Analysis and Optimization of Hybrid Software-Defined Networks

Von der Fakultät für Elektrotechnik, Informationstechnik, Physik
der Technischen Universität Carolo-Wilhelmina zu Braunschweig

zur Erlangung des Grades eines Doktors

der Ingenieurwissenschaften (Dr.-Ing.)

genehmigte Dissertation

von:	Marcel Caria
aus:	Braunschweig
eingereicht am:	28. Oktober 2016
mündliche Prüfung am:	8. Februar 2017

Referent:	Prof. Admela Jukan, TU Braunschweig
Referent:	Prof. Xavier Masip Bruin, UPC Barcelona
Referent:	Dr. Marco Hoffmann, NOKIA Bell Labs München
Vorsitzender:	Prof. Thomas Schneider, TU Braunschweig

Druckjahr: 2017

Abstract

Hybrid IP networks that use both control plane paradigms – distributed and centralized – promise the best of two worlds: programmability and flexible control of Software-Defined Networking (SDN), and at the same time the reliability and fault tolerance of distributed routing protocols like Open Shortest Path First (OSPF). Hybrid SDN/OSPF networks typically deploy OSPF to assure carefree operation of best effort traffic, while SDN can control prioritized traffic. This "ships-passing-in-the-night" approach, where both control planes are unaware of each other's configurations, only require hybrid SDN/OSPF routers that can participate in the domain-wide legacy routing protocol and additionally connect to a central SDN controller. This mode of operation is however known for a number of challenges in operational networks, including those related to network failures, size of forwarding tables, routing convergence time, and the increased complexity of network management.

There are alternative modes of hybrid operation that provide a more holistic network control paradigm, either through an OSPF-enabled SDN controller, or a common network management system that allows the joint monitoring and configuration of both control planes, or via the partitioning of the legacy routing domain with SDN border nodes. The latter mode of operation offers to some extent to steer the working of the legacy routing protocol inside the sub-domains, which is new. The analysis, modeling, and evaluative comparison of this approach called SDN Partitioning with other modes of operation is the main contribution of this thesis.

This thesis addresses important network planning tasks in hybrid SDN/OSPF networks and provides the according mathematical models to optimize network clustering, capacity planning, SDN node placement, and resource provisioning for a fault tolerant operation. It furthermore provides the mathematical models to optimize traffic engineering, failure recovery, reconfiguration scheduling, and traffic monitoring in hybrid SDN/OSPF networks, which are vital network operational tasks.

Kurzfassung

Hybride IP-Netzwerke, die beide Control-Plane-Paradigmen einsetzen – verteilt und zentralisiert – versprechen das Beste aus beiden Welten: Programmierbarkeit und flexible Kontrolle des Software-Defined Networking (SDN) und gleichzeitig die Zuverlässigkeit und Fehlertoleranz von verteilten Routingprotokollen wie Open Shortest Path First (OSPF). Hybride SDN/OSPF-Netze nutzen typischerweise OSPF für die wartungsarme Bedienung des Best-Effort-Datenverkehrs, während SDN priorisierte Datenströme kontrolliert. Bei diesem Ansatz ist beiden Kontrollinstanzen die Konfiguration der jeweils anderen unbekannt, wodurch hierbei hybride SDN/OSPF Router benötigt werden, die am domänenweiten Routingprotokoll teilnehmen können und zusätzlich eine Verbindung zu einem SDN-Controller herstellen. Diese Arbeitsweise bereitet jedoch bekanntermaßen eine Reihe von Schwierigkeiten in operativen Netzen, wie zum Beispiel die Reaktion auf Störungen, die Größe der Forwarding-Tabellen, die benötigte Zeit zur Konvergenz des Routings, sowie die höhere Komplexität der Netzwerkadministration.

Es existieren alternative Betriebsmodi für hybride Netze, die einen ganzheitlicheren Kontrollansatz bieten, entweder mittels OSPF-Erweiterungen im SDN-Controller, oder mittels eines übergreifenden Netzwerkmanagementsystems, dass das Monitoring und die Konfiguration aller Netzelemente erlaubt. Eine weitere Möglichkeit stellt das Clustering der ursprünglichen Routingdomäne in kleinere Subdomänen mittels SDN-Grenzknoten dar. Dieser neue Betriebsmodus erlaubt es zu einem gewissen Grad, die Operationen des Routingprotokolls in den Subdomänen zu steuern. Die Analyse, Modellierung und die vergleichende Evaluation dieses Ansatzes mit dem Namen SDN-Partitionierung und anderen hybriden Betriebsmodi ist der Hauptbeitrag dieser Dissertation.

Diese Dissertation behandelt grundlegende Fragen der Netzplanung in hybriden SDN/OSPF-Netzen und beinhaltet entsprechende mathematische Modelle zur Optimierung des Clusterings, zur Kapazitätsplanung, zum Platzieren von SDN-Routern, sowie zur Bestim-

mung der notwendigen Ressourcen für einen fehlertoleranten Betrieb. Desweiteren enthält diese Dissertation Optimierungsmodelle für Traffic Engineering, zur Störungsbehebung, zur Ablaufplanung von Konfigurationsprozessen, sowie zum Monitoring des Datenverkehrs in hybriden SDN/OSPF-Netzen, was entscheidende Aufgaben der Netzadministration sind.

Contents

List of Figures

List of Tables

1

Introduction

The routing of traffic in the backbone networks of the Internet is commonly based on distributed routing protocols like Open Shortest Path First (OSPF) or Intermediate System to Intermediate System (IS-IS). If an administrator uses one of these so-called link-state protocols, it has to be operated on all routers of the routing domain. A protocol-conform router implements Dijkstra's shortest path algorithm and uses a flooding mechanism to exchange topological (i.e. link state) information with all other routers in the network. Each router then generates its own topological map in the form of a *routing table* from the flooded information to determine its own *forwarding table*. This forwarding table assigns outgoing ports to reachable subnetworks, which in turn are identified by a range of IP addresses. These protocols decide on the routing in the network in an autonomous fashion except for a cost metric, which can be configured by the network administrator on a per-link basis to let the routing algorithm prefer (or ignore) individual links, thus, these protocols actually provide least cost and not shortest path routing. OSPF and IS-IS have proven to work consistently and predictably in the IP layer of the Internet and have demonstrated their reliable operation over a long period of time.

Software-Defined Networking (SDN), on the contrary, is a new networking paradigm based on a logically centralized and programmable control plane, that has recently gained a lot of attention. The initial intention for SDN was the separation of the control and the data plane, which are connected via a well-defined interface, i.e. between the network devices and the SDN controller. The most common implementation of this interface is called OpenFlow, which became the de-facto SDN standard due to its wide dissemination. The forwarding table of an OpenFlow-conform device is referred to as flow table and it provides far more sophisticated rules for packet handling compared to the common IP forwarding capabilities of IP routers. Each entry of a flow table matches a subset of the traffic (i.e. a "flow") and determines a certain action (e.g. forwarding on a specific port, dropping the packet, modifying specific header fields, etc.) to be performed on according packets. Popular SDN controller implementations like OpenDaylight and Floodlight provide an open northbound interface that allows to steer, to monitor, and to program all network operations by network management applications. SDN recently also became fashionable for wide-area backbone scenarios, which let most telecommunication network equipment vendors – in case they haven't already released so-called SD-WAN (i.e. Software-Defined Wide Area Network) devices – announce the intention to build OpenFlow-capable products [1].

Upgrading a telecommunication network to a fully SDN-enabled operation is however not without issues and new costly investments. In fact, ISPs are still reluctant regarding the change of the control plane paradigm in their networks from *distributed* to *centralized*, as distributed routing protocols operate consistently and predictably over years, efficiently control real life conditions, and reliably recover from network failures. A migration to SDN requires new hardware, new tools, and new expertise for network administrators, while SDN still fights with some hard-to-kill scalability preconceptions [2].

The term *hybrid control plane* refers in this context to a network architecture, where both control plane paradigms – the logically centralized SDN and a distributed routing protocol – are deployed in the same routing domain [3, 4]. While the discussion on centralized *versus* distributed network control planes is lively and ongoing in the networking community, the need for a *hybrid* networking paradigm, which can combine the advantages of both, has been broadly recognized, not least as it provides the only pragmatic migration path to SDN without the expensive requirement of replacing all legacy equipment simultaneously. Moreover, such an architecture allows for a smooth and total cost of ownership optimized migration to SDN. The most new SDN devices support some sort of a hybrid mode to allow for a gradual network upgrade. In the current approaches, hybrid SDN routers build their regular forwarding tables from OSPF, while the SDN controller can insert higher priority rules (also with more sophisticated matching parameters). Thus, hybrid control plane architectures typically use the distributed legacy routing protocol for best effort packet forwarding, while the SDN controller can inject high priority rules on top for advanced routing configurations.

This common mode of operation of hybrid control plane architectures resembles a "ships-passing-in-the-night" strategy, where both control planes are oblivious to what the other one configures in the network. This is known to create a number of challenges in an operational network:

- The uncorrelated control planes can cause forwarding anomalies like routing loops and black holes [5] in case of a network failure.

- Hybrid routers require larger forwarding tables to hold entries from both control planes, which is inefficient regarding the expensive ternary content-addressable memory (TCAM) (which is used to perform one-clock-cycle memory look-ups), power

consumption, and the required silicon space [6].

- The overall routing convergence time can increase significantly, as the SDN controller must wait for the legacy routing protocol to converge, before appropriate SDN actions can be determined.

- The network management system must consider the effect of operations, administration and maintenance (OAM) actions in both control planes, which increases the complexity of network management.

- In order to provide the maximum on network control to the central controller for a given number of SDN-enabled routers, a placement optimization for the SDN routers becomes critical.

A more sophisticated approach of a hybrid SDN/OSPF control plane can address a few of the named issues, which requires that the central SDN controller operates the legacy routing protocol in place of the actual SDN routers. These SDN routers do therefore not require any legacy protocol implementation, as they only need to forward all protocol messages to the central SDN controller. The neighborhood adjacency which a legacy router assumes to build with its SDN neighbor is thus actually initialized with a protocol instance of the legacy routing service inside the SDN controller. This in turn lets the controller be aware of the current routing configuration state, which also allows to react more adequately to network failures.

This thesis proposes *Centrally Partitioned Distributed Routing Domains* as the operational mode and new architecture for hybrid networks, or for short *SDN Partitioning (SDNp)*. In this approach, SDN switches are used as border nodes to *partition* the original distributed (e.g., OSPF) routing domain into sub-domains. With OSPF, for instance, the SDN nodes *appear* to their legacy neighbors as regular OSPF routers, while they actually act as simple protocol repeaters

that forward all OSPF messages to the centralized SDN controller, where protocol messages can be modified before they are returned to the sending node and then flooded across the sub-domain border. This in turn allows to reconfigure the routing of traffic *between* sub-domains by determining the exit border node on a per-destination basis. In this scheme, the distributed routing protocol remains stable at all times, while inter-sub-domain routes (which contribute the majority of traffic) are controlled in a centralized fashion. This thesis details the network architecture with SDNp, and it provides the complete mathematical background of the scheme, including the theory and complexity of Link State Advertisements (LSA) generation. In addition, it provides a partitioning method that generalizes a prominent model for the vertex separator problem.

This thesis analyzes and compares common approaches of hybrid control plane architectures with SDNp and provides mathematical models for typical network optimization tasks in the context of telecommunication network management, e.g. capacity planning, traffic engineering, failure recovery, etc. The main contribution of this thesis is the analysis and mathematical modeling of SDNp. The provided numerical analysis shows that the performance of SDNp ranges from at least significant improvements compared to regular OSPF, up to capabilities comparable to full SDN deployment. The results also indicate that the proposed scheme is a pragmatic and efficient migration solution for SDN, that initially requires only a few SDN nodes and partitions, whereby it allows for iterative upgrades in later migration stages by further partitioning into smaller sub-domains.

1.1 Thesis Contributions

The work in this thesis contributes to a number of modeling and systems aspects of hybrid SDN/Legacy network architectures with a focus on the novel operational scheme SDNp. The primary contributions focus on accurately modeling the behavior of the IP layer that deploys both control plane schemes in parallel, the centralized SDN controller and the distributed scheme of routing protocols like OSPF. The evaluation of the proposed models numerically compares the different hybrid SDN/OSPF control plane schemes with plain OSPF and full SDN deployment.

1.1.1 Network Planning Models for Hybrid SDN/OSPF

The optimization of network resource deployment is referred to as network planning, which comprehends such vital tasks like link capacity provisioning, network clustering, the scheduling and placement of technological upgrades (i.e. migration), or the resource provisioning to assure a fault tolerant operation. This thesis therefore provides the required mathematical models for all major offline planning operations for the IP layer, which allow for determining the dimensions and capabilities of its resources, such as link capacities and operational features of the routers. Based on these models, the thesis analyzes and evaluates the performance characteristics and efficiency of the different hybrid control plane approaches with an "OSPF-only" (i.e. legacy) network and with full SDN deployment. The network planning optimization models for hybrid legacy/SDN networks presented in this thesis are new and allow for an optimal network clustering, optimal SDN node placement for hybrid SDN/OSPF operation, link capacity planning, and the minimization of the required networking resources for fault tolerant operation. The resource deployment strategies for traffic monitoring, that are proposed in this thesis, were discussed in a paper submitted to IEEE

Transactions on Network and Service Management for publication (listed as first journal paper in Subsection 1.2). The submitted paper is a major extension of one of our previous publications (listed as seventh conference paper in Subsection 1.2), which received the Best Paper Award at the 2013 IEEE/IFIP International Workshop on Management of the Future Internet.

This thesis proposes a novel hybrid SDN/OSPF control plane architecture named SDN Partitioning (SDNp), that partitions the legacy routing domain into sub-domain with SDN border nodes. This, in turn, provides to some extent the control over the distributed routing protocol inside the sub-domains to the centralized SDN controller by altering the routing information when it's flooded across sub-domain borders. It will be explained in detail how SDNp allows to trade off the degree of dynamic control against the autonomy and carefreeness of OSPF. We will show how to adjust the size of the sub-domains to the operator's requirements with a minimum number of SDN-enabled networking equipment. The SDN node location is however also a key characteristic for regular (non-SDNp) hybrid control schemes, as these new devices provide capabilities in terms of routing control and traffic monitoring, that might go to waste when counterproductively placed in the topology. This thesis therefore provides resource placement strategies that maximize routing control and allow to solve a traffic monitoring problem that is inherent to the IP layer. Two of the most important offline planning problems – link capacity dimensioning and the determination of the minimum provisions for a fault tolerant operation – can be solved for all the here compared control plane schemes with the novel optimization models provided in this thesis.

1.1.2 Network Operational Models for Hybrid SDN/OSPF

The optimal management and operation of network resources requires an exact mathematical modeling of the routing capabilities

and constraints of the given network architecture. Such models are here provided for the operation of different hybrid SDN/OSPF network architectures, which are the second major contribution of this thesis. These models allow for a detailed analysis and comparison of fundamental network operational tasks. Traffic engineering is one of these tasks that have to date not been modeled comprehensively for hybrid SDN/OSPF architectures. The mathematical model of failure recovery provided in this thesis is closely related to the problem of traffic engineering and therefore requires only an extension of the previous one that allows to take routing stability into consideration. The provided models are very efficient and thus solve comparably quickly due to the practice of pre-computing all possible valid routing paths before the optimization, which significantly reduces the models' complexity.

This thesis also provides a scheduling scheme for frequent network reconfigurations, e.g. for load balancing or load adaptive energy saving operations. It will be shown how the optimal timing of such operations can reduce the total number of reconfigurations, which is preferred by network administrators to support trouble-free network operations. This thesis finally proposes how to support traffic monitoring with throughput calculations based on easily accessible byte counters in SDN routers, or alternatively on backup links of legacy routers. The problem is addressed in detail: Timing issues of the involved management protocol, a proof-of-concept implementation considering the aspects and advantages of Network Function Virtualization, and hands-on experiences in the testbed lab at TU Braunschweig are discussed, before we numerically evaluate the proposed monitoring scheme in terms of resource requirements.

1.1.3 Linear Optimization, Implemented Software, Models, and Publications

All results shown in this thesis were computed on an Intel Core i7-3930K CPU (6 x 3.2 GHz), and we used the GUROBI optimizer [7] to solve all ILP-based problems. The comprehensive simulation framework for hybrid SDN/OSPF networks, that was developed in the course of the work presented in this thesis, as well as the implementation of the shown proof-of-concept traffic monitor, and all discussed optimization models (which implement the API of the GUROBI optimizer), will not be further discussed in this thesis. However, all implementations are provided open-source as Java code on the public repository hosting system GitHub at: `https://github.com/marcel-caria/SDNp.git`. The most important publications in this thesis have been made available at the arXiv (`http://arxiv.org/`), which is the open preprint repository for scientific papers at the Cornell University.

1.2 Supporting Publications

1.2.1 Journal Articles

1. M. Caria and A. Jukan, "On the IP traffic matrix problem in hybrid SDN/OSPF networks," *submitted for review to IEEE Transactions on Network and Service Management.* (Preprint available at arXiv.org: `https://arxiv.org/abs/1610.08256`)

2. M. Caria, A. Jukan, and M. Hoffmann, "SDN Partitioning: A centralized control plane for distributed routing protocols," *IEEE Transactions on Network and Service Management,* Volume 13, Number 3, pp. 381-393, September 2016. (Preprint available at arXiv.org: `https://arxiv.org/abs/1604.04634`)

3. M. Yannuzzi, M. Siddiqui, A. Sällström, B. Pickering, R. Serral-

Gracià, A. Martínez, W. Chen, S. Taylor, F. Benbadis, J. Leguay, E. Borrelli, I. Ormaetxea, K. Campowsky, G. Giammatteo, G. Aristomenopoulos, S. Papavassiliou, T. Kuczynski, S. Zielinski, J. Seigneur, C. B. Lafuente, J. Johansson, X. Masip-Bruin, M. Caria, J. R. Junior, E. Salageanu, and J. Latanicki, "Tefis: A single access point for conducting multifaceted experiments on heterogeneous test facilities," *Computer Networks,* Volume 63, pp. 147-172, 2014, Special Issue on Future Internet Testbeds Part II.

4. M. Caria, M. Chamania, and A. Jukan, "A comparative performance study of load adaptive energy saving schemes for IP-over-WDM networks," *IEEE/OSA Journal of Optical Communications and Networking,* Volume 4, Number 3, pp. 152-164, March 2012.

5. M. Chamania, M. Caria, and A. Jukan, "Achieving IP routing stability with optical bypass," *Optical Switching and Networking,* Volume 7, Number 4, pp. 173-184, December 2010.

1.2.2 Conferences and Workshops

1. M. Caria and A. Jukan, "Link capacity planning for fault tolerant operation in hybrid SDN/OSPF networks," *accepted for publication at GLOBECOM 2016*, Washington, USA, December 2016. (Preprint available at arXiv.org: `https://arxiv.org/abs/1604.05534`)

2. M. Caria and A. Jukan, "The perfect match: Optical bypass and SDN Partitioning," in *2015 IEEE 16th International Conference on High Performance Switching and Routing (HPSR),* July 2015, pp. 1-6.

3. T. Das, M. Caria, A. Jukan, and M. Hoffmann, "Insights on

SDN migration trajectory," in *2015 IEEE International Conference on Communications (ICC),* June 2015, pp. 5348-5353.

4. M. Caria, T. Das, and A. Jukan, "Divide and Conquer: Partitioning OSPF networks with SDN," in *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM),* May 2015, pp. 467-474. (Preprint available at arXiv.org: `https://arxiv.org/abs/1410.5626`)

5. M. Caria, F. Carpio, A. Jukan, and M. Hoffmann, "Migration to energy efficient routers: Where to start?," in *Communications (ICC), 2014 IEEE International Conference on,* June 2014, pp. 4300-4306.

6. M. Caria, A. Jukan, and M. Hoffmann, "A performance study of network migration to SDN-enabled traffic engineering," in *Global Communications Conference (GLOBECOM), 2013 IEEE,* December 2013, pp. 1391-1396.

7. (Best Paper Award) M. Caria and A. Jukan, "A novel approach to accurately compute an IP traffic matrix using optical bypass," in *2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013),* May 2013, pp. 1135-1141.

8. M. Caria, A. Engelmann, A. Jukan, and B. Konrad, "How to slice the day: Optimal time quantization for energy saving in the Internet backbone networks," in *Global Communications Conference (GLOBECOM), 2012 IEEE,* December 2012, pp. 3122-3127.

9. M. Caria, M. Chamania, and A. Jukan, "To switch on or off: A simple case study on energy efficiency in IP-over-WDM networks," in *High Performance Switching and Routing (HPSR),*

2011 IEEE 12th International Conference on, July 2011, pp. 70-76.

10. M. Chamania, M. Caria, and A. Jukan, "A comparative performance analysis of IP traffic offloading schemes over dynamic circuits," in *INFOCOM, 2011 Proceedings IEEE,* April 2011, pp. 136-140.

11. M. Caria, M. Chamania, and A. Jukan, "Trading IP routing stability for energy efficiency: A case for traffic offloading with optical bypass," in *Optical Network Design and Modeling (ONDM), 2011 15th International Conference on,* February 2011, pp. 1-6.

12. M. Chamania, M. Caria, and A. Jukan, "Effective usage of dynamic circuits for IP routing," in *Communications (ICC), 2010 IEEE International Conference on,* May 2010, pp. 1-6.

13. M. Chamania, M. Caria, and A. Jukan, "Achieving IP routing stability with optical bypass," in *Proceedings of the 3rd International Conference on Advanced Networks and Telecommunication Systems,* ser. ANTS'09. Piscataway, NJ, USA: IEEE Press, 2009, pp. 43-45.

1.3 Thesis Organization

This thesis is structured in five chapters. After this introductory chapter, Chapter 2 explains the network architecture of legacy IP networks and hybrid SDN/OSPF networks in general, and of SDN Partitioning in particular. The third chapter presents the modeling of common network optimization tasks from the area of network planning in hybrid SDN/OSPF networks, including a numerical analysis comparing the performance of common hybrid architectures with SDN Partitioning and traditional legacy routing schemes.

Chapter 4 provides the same for network operational and management tasks like traffic engineering, failure recovery, configuration scheduling, and traffic monitoring, that have to date not been addressed adequately in the context of hybrid SDN/OSPF networks in the literature. Finally, the fifth chapter concludes the thesis.

2
Network Architecture

2.1 Introduction

This chapter explains the network architecture, on which the analysis and modeling in the following chapters is based on. As the majority of contributions in this thesis is related to the IP layer of Internet backbone networks, this chapter starts with a detailed description of the two fundamental control plane schemes, distributed routing protocols vs. Software-Defined Networking. The following section explains how the two schemes can work together in the same network, which is referred to as hybrid SDN/OSPF, and an own section is dedicated to explaining the mechanisms, constraints, and particularities of the operational scheme SDN Partitioning, which is in the focus of this thesis. Finally, the made assumptions regarding Layer 2 and its interplay with the IP layer's control plane, and details on the energy management of network resources are explained.

2.2 Supporting Publications

1. M. Caria, A. Jukan, and M. Hoffmann, "SDN Partitioning: A centralized control plane for distributed routing protocols," *IEEE Transactions on Network and Service Management,* Volume 13, Number 3, pp. 381-393, September 2016. (Preprint

available at arXiv.org: https://arxiv.org/abs/1604.04634)

2. M. Caria, M. Chamania, and A. Jukan, "A comparative performance study of load adaptive energy saving schemes for IP-over-WDM networks," *IEEE/OSA Journal of Optical Communications and Networking,* Volume 4, Number 3, pp. 152-164, March 2012.

3. M. Caria, F. Carpio, A. Jukan, and M. Hoffmann, "Migration to energy efficient routers: Where to start?," in *Communications (ICC), 2014 IEEE International Conference on,* June 2014, pp. 4300-4306.

4. M. Caria, A. Jukan, and M. Hoffmann, "A performance study of network migration to SDN-enabled traffic engineering," in *Global Communications Conference (GLOBECOM), 2013 IEEE,* December 2013, pp. 1391-1396.

2.3 Legacy Routing Protocols vs. Software-Defined Networking

The routing paths of IP packets in a single routing domain, i.e., in the network of a single Internet service provider (ISP), are commonly configured by a distributed algorithm executed in all routers. Open Shortest Path First (OSPF) and Intermediate System to Intermediate System (IS-IS) are so called link state routing protocols based on Edsger W. Dijkstra's shortest path algorithm and have been standardized by the IETF in their currently used versions already in 1998 (OSPF v2, RFC 2328 [8]) and 1990 (OSI IS-IS, RFC1142 [9]) respectively. Both protocols are recognizably similar in function and mechanism, but their terminology slightly differs. Please note that for the sake of simplicity and because OSPF is prevalent, we use only the OSPF terminology in this thesis. The routing updates of OSPF, which are required in the initial operational phase of the

network and after every topological change, are referred to as Link State Advertisements (LSA), and a router participating in the protocol distributes all its topological information based on a flooding mechanism throughout the entire network. All received LSAs are stored in a router's Link State Data Base (LSDB) and the mechanism that is performed after receiving an LSA is called *synchronization of databases* [8]: After header processing, the packet is tested whether the type is known and the checksum is OK. Then the own database is checked whether the LSA exists already. If this is not the case, or if the sequence number of the received is higher than in the maintained version, the received LSA is added to the database (or replaces the maintained one respectively). This action is followed by the reply with an acknowledgment, and finally triggers the transmission of the LSA on all ports but the receiving one (called *flooding*). The routing protocol has *converged* in the network, when all topological information was distributed with LSAs throughout the entire network, and all participating network nodes have calculated all routing paths.

Software-Defined Networking, in contrast to distributed routing protocols like OSPF, is a networking paradigm based on a logically centralized and programmable control plane, that has gained a lot of attention during recent years, and most network equipment vendors have announced the intention to build SDN enabled devices or have already released according products. The Open Networking Foundation is an nonprofit consortium founded for development and standardization of SDN, and a quick glance at the list of its members [1] documents SDN's relevance in the community. The basic idea of Software-Defined Networking is to decouple the control plane from the data plane to allow the centralization of all network control functions. This brings many advantages, which makes it highly desirable for network operators and researchers:

- By using a standardized interface between the control and data

plane – OpenFlow is the first viable approach in this regard – SDN allows the network to become vendor agnostic, so that an operator can mix low-cost off-the-shelf hardware from different suppliers.

- Due to the separation of the control and data plane, SDN allows independent innovation of the two and gives the research community the opportunity to easily test new ideas in operational networks. This separation furthermore provides a better scalability in data centers, as the number of instances of control and data plane elements can individually be adapted to actual work loads.

- The centralized software based network control allows for great flexibility and programmability of the network and allows the operator to customize the network according to his own preferences, e.g., elimination of unneeded features, network virtualization, etc.

A concern often raised over SDN is the scalability of the approach, assuming that a central controller can not scale with a growing network. However, recent research showed that these concerns stem from the historical evolution of SDN (e.g., poor implementation of early SDN controllers) and it was shown that there is no inherent scalability bottleneck to SDN [2].

2.4 Hybrid OSPF/SDN

The term *hybrid control plane* refers to an increasingly important network architecture, where both control plane paradigms – the logically centralized SDN and a distributed routing protocol like OSPF – are deployed in the same routing domain [3, 4]. While the discussion on centralized *versus* distributed network control planes (e.g., [10, 11]) is lively and ongoing in the networking community, the

need for a *hybrid* networking paradigm, which can combine the advantages of both, has been broadly recognized [12, 13, 14, 15, 5], not least as it provides the only pragmatic migration path to SDN without the expensive replacement of all legacy equipment. Moreover, such an architecture allows for a smooth and total cost of ownership optimized migration to SDN. In fact, many new Internet routers are equipped with an interface for OpenFlow (which is the de facto messaging standard between network devices and SDN controllers) and support a hybrid OpenFlow/OSPF mode. Hybrid control plane architectures typically use the distributed legacy routing protocol for best effort packet forwarding, while the SDN controller injects high priority rules on top for advanced routing configurations.

A typical hybrid SDN operation follows a "ships-passing-in-the-night" strategy, whereby distributed legacy routing and SDN control paradigms are oblivious to what the other one configures. This is known to create a number of challenges in an operational network, including those related to network failures, size of forwarding tables, routing convergence time, thus impeding the chances for SDN to be deployed in carrier networks. In case of network failures, for instance, the uncorrelated control planes may cause forwarding anomalies, like routing loops and black holes [5]. The size of the router's forwarding information base (FIB) is also an issue, as routers use TCAM to perform memory lookups in one clock cycle, which has to be dimensioned economically due to cost, power consumption, and the required silicon space [6]. A hybrid router, in fact, contains both the OSPF *and* OpenFlow forwarding tables, which increases the required FIB size significantly. Finally, hybrid SDN networks require optimization of the SDN router location, or else their advantages become limited.

Hybrid SDN networking has been analyzed and explained to a great extent in [12, 13, 14]. In a hybrid network, the capability of the SDN controller to insert higher priority rules into the forward-

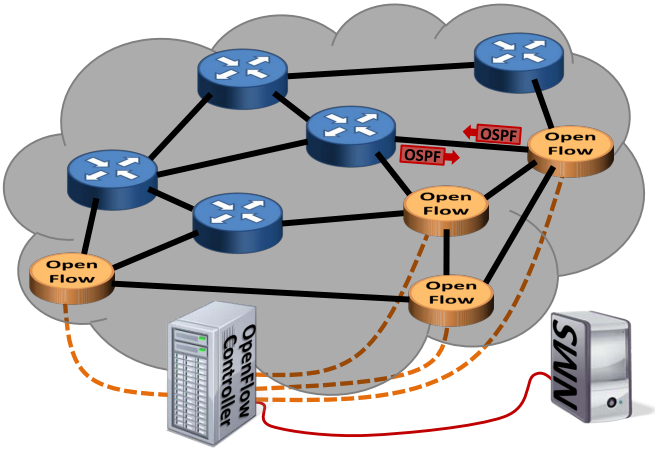


Figure 2.1: An IP network with a hybrid SDN/OSPF control plane.

ing tables is a powerful new feature which in [16] has been coined as "policy based routing on steroids". We therefore refer to this control plane approach as the *stacked hybrid* model in this thesis. A known practical implementation of a hybrid control plane is Google's *B4* [17].

The reference network architecture for hybrid SDN/OSPF operation is illustrated in Figure 2.1, showing a network with five OSPF routers and four SDN routers. After the initial migration, the conventional routers with a legacy routing protocol (i.e., OSPF) are deployed together with SDN-enabled routers. As it can be seen, all SDN routers are controlled by a centralized controller, which in turn is managed by the Network Management System (NMS). Please note that SDN alone does not provide network optimization capabilities like discussed in Chapters 3 and 4 in this thesis. Optimizations for network operations and offline planning still remain implemented in the NMS, since an optimization application typically also requires

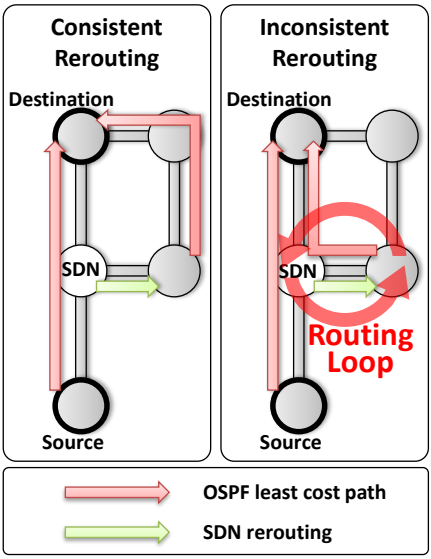


Figure 2.2: SDN rerouting and how it may lead to routing loops.

input from the traffic monitoring system, network policies from the operator, etc. However, what SDN does provide is a thorough configurability of the routing of all flows in the network, and thus a much greater solution space for network optimization, which in turn leads to an overall better performance of the network optimizations carried out by the NMS.

As not all routers in a hybrid SDN/OSPF network are dynamically configurable by the central controller, packets must always follow OSPF's least cost path to the destination, unless the packet traverses an SDN router, from where it can be forwarded on any outgoing port. This SDN reconfiguration of paths is however constraint in order to avoid routing anomalies, like depicted in Figure 2.2. The figure shows a valid rerouting from the original OSPF least cost path

at the SDN router, and a rerouting at the SDN router that leads to a routing loop, as the OSPF router to which the SDN router forwards packets to the destination has to send them back directly due to its own least cost path traversing the said SDN router.

2.5 SDN Partitioning

To address the challenges of a hybrid SDN control plane, we propose *Centrally Partitioned Distributed Routing Domains* as the operational mode and new architecture for hybrid networks, or for short *SDN Partitioning* or SDNp. In our approach, SDN switches are used as border nodes to *partition* the original distributed (e.g., OSPF) routing domain into sub-domains. With OSPF, for instance, the SDN nodes *appear* to their legacy neighbors as regular OSPF routers, while they actually act as simple protocol repeaters that forward all OSPF messages to the centralized SDN controller, where protocol messages can be modified before they are returned to the sending node and then flooded across the sub-domain border. This in turn allows to reconfigure the routing of traffic *between* sub-domains by determining the exit border node on a per-destination basis. In our scheme, the distributed routing protocol remains stable at all times, while inter-sub-domain routes (which contribute the majority of traffic) are controlled in a centralized fashion. This thesis details the network architecture with SDN Partitioning, and provides the complete mathematical background of the scheme, including the theory and complexity of LSA generation and the optimization models for typical network management tasks (i.e., traffic engineering, capacity planning, etc.) as well as the used network partitioning method that generalizes a prominent model for the vertex separator problem. Our numerical analysis shows that, in all evaluated measurements, the performance of SDN Partitioning ranges from significant improvements to regular OSPF (in its minimum configuration

with only two sub-domains), up to network control capabilities comparable to full SDN deployment (with a partitioning into smaller sub-domains).

Figure 2.3 illustrates the idea of SDN Partitioning: SDN-enabled Internet routers replace legacy routers at some strategic locations. As it is well known, mesh topologies can be partitioned in various ways: the more SDN routers in the network the larger the number of sub-domains. We will go into details of network partitioning in the next section. Let us assume for the time being that the best possible partitioning method was used and as a result the distributed routing domain in Figure 2.3 has been partitioned in three sub-domains: out of 14 legacy routers, three were replaced by SDN enabled routers, and the rest of the legacy routers are now associated with the newly created distributed routing sub-domains.

The network deploys conventional IP routers that run OSPF, while three nodes at specific locations have been replaced with OpenFlow switches that establish individual control channels to the SDN controller through their OpenFlow interfaces, and then act as border nodes to the OSPF partitions (or, sub-domains). Note that in our network, we do not deploy typical *hybrid routers*, i.e., capable of both OSPF and OpenFlow simultaneously (like [3]). Instead, these are standard OpenFlow switches. The compatibility of SDN switches with legacy routers in our operational scheme is provided by the fact that SDN switches *act* as legacy routers and respond with OSPF-conform protocol messages, as generated by the SDN controller. The SDN controller proactively configures the flow tables of all SDN switches such that they can process all user traffic, whereas all received OSPF routing protocol messages are (instead of being flooded to the opposite sub-domain) transferred to the central SDN controller. The SDN controller can in turn uses this information to enrich its own global view on the current state of the network. The response messages are in fact not generated by the SDN switches

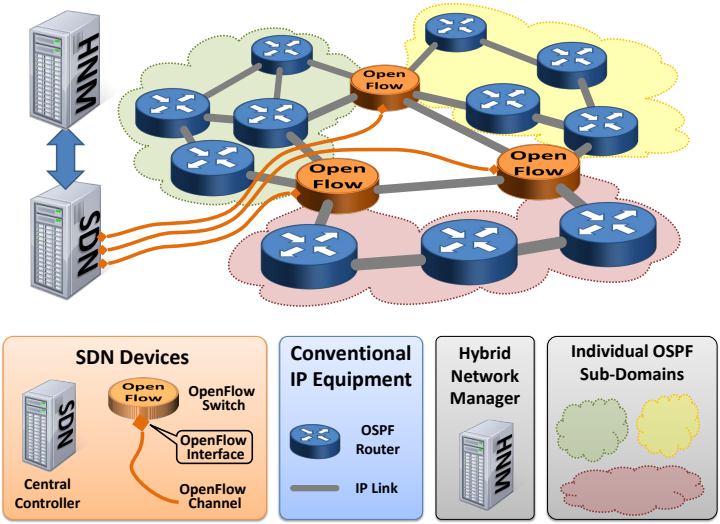


Figure 2.3: The network architecture of an SDN-partitioned OSPF domain.

(which act solely as packet forwarding switches), but by the SDN controller, before flooding is continued on the other side.

In this architecture, legacy routers are neither aware of the existence of the hybrid control plane, nor of the fact that they belong to some sub-domain. Moreover, all legacy routers connecting directly to SDN border nodes do not recognize any difference to normal OSPF routers. Across these particular links, the *OSPF neighbor adjacency*[1] is actually formed between the OSPF router and the central SDN controller *through* the SDN switch. This mechanism remains transparent for OSPF, such that the SDN switch is nonetheless con-

[1] Directly connected OSPF routers use OSPF's Hello protocol to form neighbor adjacencies, which is the prerequisite for any further protocol interaction.

sidered as regular OSPF router by its legacy neighbors. The control over the routing (of inter-sub-domain traffic) can now be achieved through the controller's manipulation of the global topological view. Please note that this operational scheme requires the implementation of the used legacy protocol in the central SDN controller (which to date is not standard), and to form neighbor adjacencies to all legacy nodes adjacent to all SDN nodes. Figure 2.3 also depicts a *Hybrid Network Manager* connecting to the SDN controller, which represents the implementation of various routing optimization schemes through various functional components, as well as the implementation of regular network management functions (e.g., monitoring, service provisioning, etc.).

2.5.1 Differentiation of SDNp from similar approaches

In regard to the network partitioning of SDNp, our method sets the goals similar to the partitioning of an OSPF domain into areas (defined in RFC 2328 [8]), where OSPF areas are used to simplify the administration of large topologies and to reduce the amount of protocol traffic. However, SDN Partitioning allows SDN-based traffic engineering by controlling the routes of inter-sub-domain traffic, which is not provided through the partitioning into OSPF areas. Also, partitioning a network into *individual OSPF domains* and connecting them with the Border Gateway Protocol (BGP) [18] would lead to a degree of freedom regarding routing control similar to SDNp. However, our method does not require the partitioning into multiple *autonomous systems* and provides a clean separation from the BGP setup that a network has already in place. Another approach partitions the network into *technology zones*, whereby each node belongs to a single zone only [19]. Our partitioning idea however is different from the zone approach, as a zone is defined as a set of interconnected nodes controlled by the same paradigm (i.e., SDN *or* OSPF). In contrary, a sub-domain in our approach is a subgraph of OSPF

routers, while the SDN nodes also participate in OSPF.

In regard to the routing protocol and architecture, SDNp has a strong relation to, but distinctive differences from the line of work published under the name "Fibbing" by Vissichio et al. in [20] and [21]. The similarity to our proposal is in the idea to introduce *fake* routing information into the OSPF routing protocol. However, the mode of operation, the requirements, and the expressiveness of our scheme and Fibbing are different. While Fibbing requires only to extend the existing network architecture with the so-called Fibbing controller (which is not SDN-related), SDNp necessitates the deployment of SDN-enabled routers. The operational differences of the two schemes are as follows: Fibbing floods fake *external* (i.e., Type 5) LSAs through the network to extend the actual topology with virtual nodes and links using the Forwarding Address field in this type of LSA, which in turn let the actual OSPF nodes recompute their shortest paths. SDNp, on the other hand, uses SDN-enabled routers to interrupt OSPF's flooding mechanism for *all* LSAs at sub-domain borders to allow the continuation of the same flooding process with customized (i.e., optimized) LSAs in the neighboring sub-domain. Though both schemes are limited by OSPF's destination-based forwarding behavior, Fibbing is slightly more expressive than SDNp, as it provides *full* control over any destination's next hop at any router, whereas SDNp preserves OSPF's control over locally limited (i.e., intra-sub-domain) traffic. By design, Fibbing uses the Forwarding Address field in Type 5 LSAs, which is considered as an exotic feature of OSPF[2]. Also, Fibbing depends on external traffic measurement tools for routing optimizations, whereas SDNp is self-contained in this regard: all flows crossing SDN-enabled routers are automatically monitored by the central controller, as per standard features of SDN, since OpenFlow uses byte counters for all flow table entries. Our routing optimizer does not need external tools and can

[2]See Cisco [22] and Juniper [23] knowledge base articles.

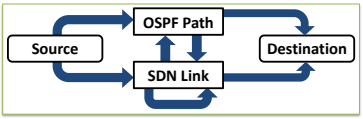


Figure 2.4: Concatenation of path elements in SDNp.

use Simple Network Management Protocol (SNMP) link counters from the inside of the sub-domains in addition to the flow counters. Finally, unlike other approaches, SDNp *decreases* protocol overhead by suppressing those LSAs at border nodes that are without an effect for the particular sub-domain.

2.5.2 Routing

In the SDNp network architecture, a valid routing path is a concatenation of OSPF paths and SDN links, as depicted Figure 2.4. Our particular notion of the terms *OSPF path* and *SDN link* differs from common usage: We define an SDN link as a directional connection between an SDN router and any other (SDN or OSPF) router. An OSPF path is defined as the *unique least cost path* between a (non-SDN) OSPF router and any other (SDN or OSPF) router *within the same sub-domain*; other protocol mechanisms like Equal Cost Multi Path are not considered.

Figure 2.5 further illustrates the routing constraints. The source node S has exactly one least cost path to each SDN border node (X and Y) in Sub-Domain 1. Thus, the route from S to D starts either with OSPF path $P1$ or $P2$, depending on the aggregated cost metric for the routing to D, i.e., the aggregated metrics along $P1$ plus the metric advertised by X for reaching D, compared to the aggregated metrics along $P2$ plus the metric advertised by Y for reaching D. The next element in the route to D is an SDN link. As an SDN node's flow table can be arbitrarily configured for all packets from

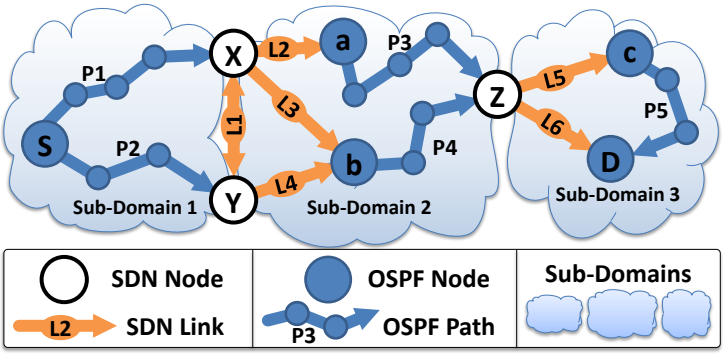


Figure 2.5: Constraints on routing in SDNp.

S to D, we see that $P1$ can be continued with SDN links $L1$, $L2$ or $L3$. In fact, $P1$ can be continued even with SDN links $L1$ and $L4$ successively. Note the self loop of the *SDN link* box in Figure 2.4: Because forwarding in SDN nodes is arbitrarily configurable and not constraint by OSPF, SDN links can be concatenated arbitrarily. Arriving at any of the two first OSPF routers (a or b) in Sub-Domain 2, there is no choice to be made, because each of the two routers has only a single OSPF path to border node Z. Finally, Z can then be configured to forward the packets directly via link $L6$ to D, or in case of congestion on that link, forward the packets via $L5$ to OSPF node c, from where the packets again have to take the regular OSPF path to the destination.

2.5.3 Network model

This subsection explains the theoretical background, problem complexity, and algorithm of one fundamental building block of SDNp, which is the routing modification via LSA generation. The here used notation is summarized in Table 2.1.

SDNp allows to advertise topology information customized per

Set	Meaning
R	Set of all *regular* (i.e. legacy OSPF) nodes
B	Set of all *border* (i.e. SDN) nodes
$\mathcal{N} = R \cup B$	Set of all nodes, with $R \cap B = \emptyset$
$\mathcal{N}_k,\ R_k,\ B_k$	Set of all, all regular, or all border nodes of the k^{th} sub-domain
$\overline{\mathcal{N}}_k$	Set of all nodes in $\mathcal{N} \backslash (R_k \cup B_k)$
M	Set of all metric vectors $\vec{m}$
E	Set of all exit vectors $\vec{e}$
Q	Set of all quantity vectors $\vec{q}$
Integer	**Meaning**
$m(n_1, n_2)$	Metric of link (n_1, n_2)
$\tilde{m}(b, d, k)$	Metric advertised by border node b for destination d in sub-domain k
$\delta(r, b)$	Metric distance between r and b
$\delta(r, b, d, \vec{m})$	Aggregated metric distance from r via b to d with advertised $\vec{m}$
Boolean	**Meaning**
$cons(p, \vec{m})$	Usage of path p is consistent with the advertisement of $\vec{m}$

Table 2.1: Parameters for SDNp

sub-domain, as illustrated in Figure 2.6: the original topology and its partitioning (shown in Subfig. a) is not exposed to the nodes in sub-domain k (i.e., nodes 1, 2, and 3). Instead, the customized view provided to these nodes pretends that both border nodes a and b have direct links to all other nodes (like shown in Subfig. b). This way, the exit node for each inter-sub-domain flow can be determined on a per-destination basis simply by setting the OSPF link weights of the virtual connections. For instance, setting x and y determines how traffic for d exits sub-domain k. Please note that the OSPF

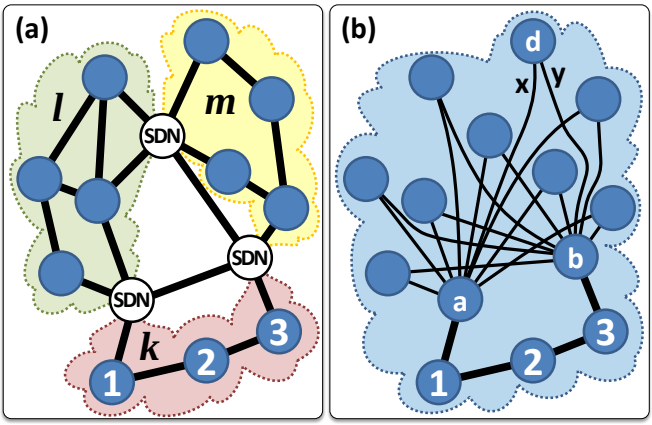


Figure 2.6: A subdomain's view on an SDN-partitioned network.

node	1	2	3	
aggregated metric to *a*	10	20	30	
aggregated metric to *b*	30	20	10	
aggregated metric to *d* via *a*	20	30	40	x = 10
aggregated metric to *d* via *b*	70	60	50	y = 40
aggregated metric to *d* via *a*	20	30	40	x = 10
aggregated metric to *d* via *b*	55	45	35	y = 25
aggregated metric to *d* via *a*	35	45	55	x = 25
aggregated metric to *d* via *b*	40	30	20	y = 10
aggregated metric to *d* via *a*	50	60	70	x = 40
aggregated metric to *d* via *b*	40	30	20	y = 10

Figure 2.7: The used exit node depends on the advertised link metrics.

nodes 1, 2, and 3 are not aware of the fact that they form a subdomain, and believe that the border SDN nodes a and b are regular OSPF neighbors.

The expressiveness of routing in SDN-partitioned networks is constrained by the requirement that the usage of routing paths must be consistent with the advertised link metrics. More precisely, the routes of all flows, which start at the OSPF nodes $r_1 \dots r_\alpha$ of the same sub-domain and that are destined for the same sub-domain-external destination d, depend on the metrics $m(b_1, d) \dots m(b_\beta, d)$ in the set of LSAs that has been advertised for d by the border (SDN) routers $b_1 \dots b_\beta$ of that sub-domain. It can be seen in Figure 2.6 that traffic flows, which enter the network through a node of sub-domain k and exit the network, for example, through node d, leave sub-domain k either via node a or b, depending on the aggregated OSPF link metrics to the border nodes plus the metric $m(a, d)$ advertised by a and $m(b, d)$ advertised by b (denoted as x and y in Figure 2.6b) for the virtual links to d. If we assume that all links inside the sub-domain have an identical metric of 10, we can see all four possible routing scenarios in Figure 2.7. A field is marked green, if it represents the least cost path for a set of cost metrics. To give a counterexample for an impossible route combination, consider in Figure 2.6b the two routes $1 \to 2 \to 3 \to b \to d$ and $3 \to 2 \to 1 \to a \to d$. There is no set of metrics $(m(a, d), m(b, d))$ that can be advertised by border nodes a and b that would allow this combination of routes, as the use of path $1 \to 2 \to 3 \to b \to d$ presupposes

$$\begin{aligned} & m(1,2) + m(2,3) + m(3,b) + m(b,d) \\ < \; & m(1,a) + m(a,d) \qquad\qquad \Rightarrow m(b,d) < m(a,d) \end{aligned} \tag{2.1}$$

and the use of path $3 \to 2 \to 1 \to a \to d$ presupposes

$$\begin{aligned} & m(3,2) + m(2,1) + m(1,a) + m(a,d) \\ < \; & m(3,b) + m(b,d) \qquad\qquad \Rightarrow m(a,d) < m(b,d) \end{aligned} \tag{2.2}$$

which is a contradiction. Consequently, the number of actually available route combinations is assumably less than the number of possible flow / exit node combinations.

Uniqueness: A network with a set of nodes $\mathcal{N}$ contains a set of OSPF nodes R and a set of SDN border nodes B with $\mathcal{N} = R \cup B$. A network partitioned into K sub-domains is denoted as K-partitioned, and the sub-domains are ordered and numbered $1, 2, \dots K$ in a unique fashion. Also, each node $n \in \mathcal{N}$ has a network-wide unique identifier. Each node $r \in R$ belongs to a single sub-domain, and there is a subset R_k of OSPF nodes for each sub-domain k. The $|R_k| = \alpha$ nodes in R_k are ordered $R = (r_1 \dots r_\alpha)$ in a unique fashion (e.g., lexicographically regarding their ID). Each (SDN) border node $b \in B$ belongs to multiple sub-domains, but for each sub-domain k, the order of its β border nodes $B_k = (b_1 \dots b_\beta)$ is unique as well. Furthermore, we denote $\mathcal{N}_k = R_k \cup B_k$ as the set of all (i.e., domain-internal and border) nodes of sub-domain k and $\overline{\mathcal{N}}_k = N \backslash \mathcal{N}_k$ as the set of all nodes external to sub-domain k.

Distance: Each link (n_1, n_2) is assigned an integer link metric value $m(n_1, n_2)$ and $m_k(n_1, n_2)$ denotes a dynamic metric (i.e., one that can be modified by the central controller) which is advertised in sub-domain k. A metric distance $\delta(r_1, b)$ is defined as the summation of link metrics $m(r_1, r_2) + m(r_2, r_3) \dots + m(r_n, b)$ along the n hop least cost path between an OSPF node r_1 and an SDN border node b of the same sub-domain. All distances solely depend on the initial configuration of link metrics at OSPF nodes, and we assume that the configuration of those link metrics is static. We accordingly assume that distances are static for all sub-domains. Furthermore, we assume that least cost paths are unique (i.e., there exists always exactly one least cost path between any two nodes in the same sub-domain with metric distance δ) and no mechanisms like Equal Cost Multi Path are in use. Finally, as a border node b belongs to multiple sub-domains, dynamic metrics $\tilde{m}$ for a sub-domain-external destination d are advertised on a per-sub-domain basis as $\tilde{m}(b, d, k)$ and additionally indicated by the sub-domain to which they belong.

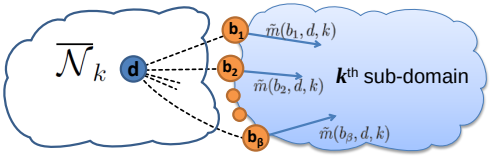


Figure 2.8: A metric vector is a set of link metrics advertised by border nodes.

Metric vector: A set of link metrics that can be advertised by the β border nodes of B_k for a destination $d \in \overline{\mathcal{N}}_k$ is denoted as metric vector $\vec{m} = (\tilde{m}(b_1, d, k), \tilde{m}(b_2, d, k), \ldots \tilde{m}(b_\beta, d, k))$. Like depicted in Figure 2.8, the components of a metric vector are ordered according to the ordering of the correspondent border nodes in that sub-domain, and the i^{th} component $\tilde{m}(b_i, d, k)$ is denoted as $\vec{m}_i$. A metric vector is *valid*, if the advertisement of its link metrics leads to a non-ambiguous single path routing scenario (i.e., without resulting in multiple least cost paths between any pair of nodes). Two metric vectors $\vec{m} \equiv \vec{m}'$ are defined to be equivalent, if they result in the same routing. In other words, if we alter $\vec{m}$ to $\vec{m}'$ such that the changes of its elements are small enough and few enough, the routing will not change and we call the vectors equivalent. The equivalence class $[\vec{m}]$ of a metric vector is the set of all metric vectors $\vec{m}'$ with $\vec{m}' \equiv \vec{m}$. As equivalence in routing means redundancy, we ignore all other elements of $[\vec{m}]$ other than a single representative element $\vec{m}$. The set of all valid, non-equivalent (i.e. representative) metric vectors in sub-domain k is denoted M_k. With OSPF-conform link metrics $1 \leq m \leq 65535$ and β border nodes in the k^{th} sub-domain, the size of the search space for a brute force approach of finding all non-equivalent metric vectors in M_k is $\mathcal{O}(65535^\beta)$, because each of the β border nodes can advertise any cost $1 \leq m \leq 65535$. This suggests the use of a more efficient search approach for M_k. Please note that we interchangeably use the terms metric vector and the

less formal but more common *LSA set*.

Exit vector: We denote the metric distance from node r via border node b to destination d with link metrics $\vec{m}$ advertised for d as $\delta(r, b, d, \vec{m})$. The exit border node $e(r, d, \vec{m}) \in B_k$ for packets from a node $r \in R_k$ to a destination $d \in \overline{\mathcal{N}}_k$ can now be determined by the used metric vector $\vec{m}$ that was advertised for d in sub-domain k as follows:

$$\begin{aligned} \forall k \in [1, K],\ \ \forall d \in \overline{\mathcal{N}}_k,\ \ \forall r \in R_k,\ \ \forall \vec{m} \in M_k : \\ e(r, d, \vec{m}) = b' \\ \text{if} \quad \delta(r, b', d, \vec{m}) = \min\{\delta(r, b, d, \vec{m}) | b \in B_k\} \end{aligned} \tag{2.3}$$

The exit vector $\vec{e}(k, d, \vec{m}) = (e(r_1, d, \vec{m}), \ldots, e(r_\alpha, d, \vec{m}))$ in sub-domain k for packets from the α nodes $r \in R_k$ to the external destination $d \in \overline{\mathcal{N}}_k$ is defined as the set of used exit border nodes (in the order according to R_k). We can shorten the notation to $e(r, \vec{m})$ and $\vec{e}(k, \vec{m})$ respectively, as the actual destination is irrelevant for generic considerations on exit nodes and exit vectors. We denote $E_k = \{\vec{e}(k, \vec{m}) | \vec{m} \in M_k\}$ as the set of all exit vectors in sub-domain k. Please note that the mapping $M_k \to E_k$ is bijective by definition, as each representative metric vector determines one unique routing scenario. It follows that the number $|E_k|$ of exit vectors in E_k equals the number $|M_k|$ of metric vectors in M_k. However, even under the strong assumption that we can find a metric vector for an exit vector in constant time, the time complexity for a brute force approach to find all exit vectors in E_k is still in $\mathcal{O}(\beta^\alpha)$ (i.e. each of the α OSPF nodes must use one of the β border nodes as exit).

Quantity vector: We define $R_k(b, \vec{m}) \subseteq R_k$ as the set that contains all domain-internal nodes of sub-domain k that use border node b as exit in case $\vec{m} \in M_k$ was advertised. If $\vec{e}(k, \vec{m})$ is given, we can determine *how many* OSPF nodes of R_k use a specific $b \in B_k$ as

exit. We denote this measure as *quantity vector*:

$$\vec{q}(k, \vec{m}) = (|R_k(b_1, \vec{m})|, \ldots, |R_k(b_\beta, \vec{m})|) \tag{2.4}$$

We denote the set of all quantity vectors in sub-domain k as $Q_k = \{\vec{q}(k, \vec{m}) | \vec{m} \in M_k\}$. The search for all $\vec{q} \in Q_k$ (i.e. finding all ways to assign the undistinguished α sub-domain-internal OSPF nodes to the β border nodes) is a combinatorics problem, whose time complexity is determined by the binomial coefficient $\binom{\beta+\alpha-1}{\alpha}$. The mapping $E_k \rightarrow Q_k$ is obviously surjective, because each element $\vec{q}(k, \vec{m}) \in Q_k$ is by definition generated by one element $\vec{e}(k, \vec{m}) \in E_k$. More interestingly, the mapping is also injective and thus has a unique inverse mapping $E_k \leftarrow Q_k$, as we can always uniquely determine an exit vector (and thus a metric vector), if the quantity vector is given, which in turn has practical relevance regarding the complexity of the computation of all M_k of a network.

Proof of injectivity: The injectivity of $E_k \rightarrow Q_k$ can be proven by contradiction, if we assume that there exists a quantity vector $\vec{q}(k, \vec{m})$ that can be generated by two different exit vectors $\vec{e}(k, \vec{m})$, $\vec{g}(k, \vec{m'})$ with $\vec{m} \not\equiv \vec{m'}$, and thus $\vec{e} \neq \vec{g}$: A quantity vector is determined by an exit vector by *counting* the occurrence of border nodes in it. It follows that if $\vec{e} \neq \vec{g}$ transform into the same quantity vector, $\vec{g}$ has to be a permutation of $\vec{e}$. In other words, the number of occurrences of each border node is the same in $\vec{e}$ and $\vec{g}$, and only their inner ordering is different. Such a permutation implies that a subset $X \in R_k$ of OSPF nodes has *swapped* their exit nodes. Exit swapping, however, can in general be described as some OSPF node r_1 initially using border node b_1 and some other OSPF node r_2 using b_2 as exit with $\tilde{m}_1$, $\tilde{m}_2$ advertised by b_1, b_2, or more formal:

$$\delta(r_1, b_1) + \tilde{m}_1 < \delta(r_1, b_2) + \tilde{m}_2 \tag{2.5}$$

and

$$\delta(r_2, b_2) + \tilde{m}_2 < \delta(r_2, b_1) + \tilde{m}_1 \tag{2.6}$$

This can be reformulated to

$$c_1 < \tilde{m}_1 - \tilde{m}_2 < c_2 \tag{2.7}$$

with constant values

$$c_1 = \delta(r_2, b_2) - \delta(r_2, b_1) < c_2 = \delta(r_1, b_2) - \delta(r_1, b_1) \tag{2.8}$$

The different metrics $\tilde{m}_3$, $\tilde{m}_4$ must now lead to the swapping of exit nodes, i.e., to

$$c_1 > \tilde{m}_3 - \tilde{m}_4 > c_2 \tag{2.9}$$

which however implies $c_1 > c_2$ and thus contradicts Eq. 2.8. □

2.5.4 LSA Generation Algorithm

The injectivity of $E_k \to Q_k$ has profound impact on the complexity of our algorithm that generates M_k in our network, as the metric vectors can be identified based on quantity vectors rather than exit vectors. A simple numerical example demonstrates the simplification: In a small sub-domain with $\beta = 4$ SDN border nodes and $\alpha = 8$ domain-internal OSPF nodes, the number of metric vectors that we would have to test regarding their non-equivalence in M_k is 65535^{β}, which is $> 10^{19}$. The number of different exit vectors we'd have to test in the same example is only β^{α}, which is 65536. Finally, the number of quantity vectors in the same example would in turn only be $\binom{\beta+\alpha-1}{\alpha}$, which is 165. The algorithm we developed to determine M_k searches for all metric vectors that generate a unique quantity vector. It can be thought of as a tally counter, which always advances the element with the smallest index i (i.e., the metric with the smallest index i in $\vec{m}$) a single step (i.e., till the point where some OSPF node switches its border node), until that element has reached the end (i.e., that metric is so large that b_i is no longer the exit node for any OSPF node). In that event, that element is turned back to zero and the subsequent element is

Input: $R_k, B_k, \delta(r,b), \vec{my}$
Output: $\vec{mz}$

```
// Step 1: Determine the component index i
for n := β to 0 do
    if mȳ_n > 0 then
        i := n;
    end
end
if i = β then
    Terminate Algorithm!
end
// Step 2: Determine value v
v := Δ_k^max;
for n := i + 1 to β do
    for q := 0 to α do
        if v > δ(r_q, b_n) + mȳ_n then
            v := δ(r_q, b_n) + mȳ_n + 1;
        end
    end
end
// Step 3: Generate mz
mz := mȳ;
mz_i := mz_i + v;
for n := i − 1 to 0 do
    mz_n := 0;
end
```

Algorithm 1: Generation of a Metric Vector

advanced, and so on. The algorithm works in detail as follows: The initial metric vector $\vec{m0} = (m_1, \ldots, m_\beta) \in M_k$ has values $m_1 = 0$ and $m_x = (x-1) \cdot (1 + \Delta_k^{\max})$ for all $2 \leq x \leq \beta$ with $\Delta_k^{\max}$ as the maximum difference between any two metric distances in subdomain k:

$$\Delta_k^{\max} = \max\{\delta(r,b) - \delta(r',b') | r, r' \in R_k,\, b, b' \in B_k\} \tag{2.10}$$

The quantity vector for this metric vector is $\vec{q} = (\alpha, 0, \ldots, 0)$, as all $|R_k| = \alpha$ OSPF nodes in R_k will use the first border node as exit. (Please note our notation: the n^{th} component of $\vec{a}$ is denoted $\vec{a}_n$ and

the i^{th} element of a set of vectors A is denoted $\vec{ai}$.) The remaining metric vectors are generated iteratively, such that the z^{th} metric vector $\vec{mz}$ is a copy of the y^{th} metric vector $\vec{my}$ (where $z = y - 1$) with its i^{th} component (i.e., metric) increased by value v as detailed in Algorithm 1. Among all representative metric vectors of M_k, this algorithm generates additionally a number of equivalent and a few non-valid ones, which are filtered out by testing for the uniqueness of the generated quantity vector.

2.6 IP-over-Optical, Optical Bypass

In order to facilitate the use of dynamic circuit services provided by the optical transport layer, we assume that each IP router is co-located with an optical network switch as shown in Fig. 2.9, which is typically the case in current carrier networks. It is assumed that the optical transport network supports both, leased-line as well as dynamic circuit services, where static IP links are established using leased-lines. In an *opaque* optical network, incoming wavelengths are terminated at each node and multi-hop transmission is facilitated via optical-electrical-optical conversions. In a *transparent* optical network, on the other hand, optical cross connects (OXC) or reconfigurable optical add-drop multiplexers (ROADM) additionally permit WDM connections between physically non-adjacent nodes. In the reminder of this thesis, we will simply use the term dynamic optical circuit to refer to either an opaque or transparent optical path, unless otherwise noted.

Optical bypass is a term associated with the phenomenon where a dynamic optical circuit (opaque, or transparent) is used to interconnect two non-neighboring IP routers. We, however, also differentiate optical bypasses based on the way they are advertised in the IP network. In conventional Network Engineering (NE) parlance, an optical bypass created between a pair of IP routers is usually ad-

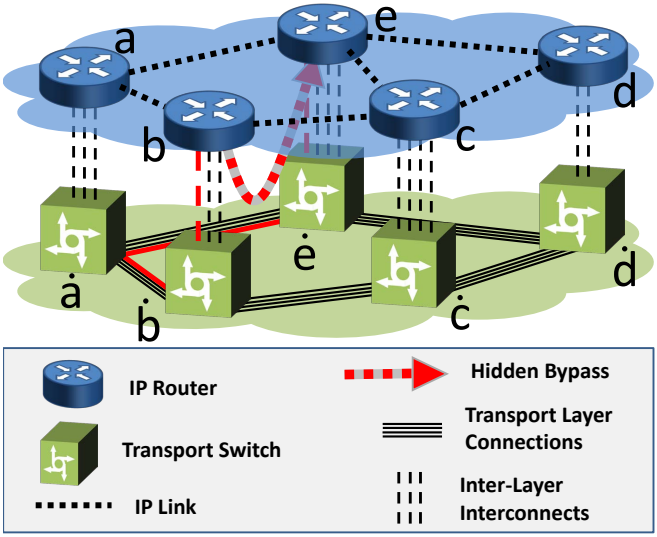


Figure 2.9: The hidden bypass.

vertised in the IP routing protocol as a new IP link. On the other hand, as proposed in [24], a multi-hop circuit setup between a pair of IP routers which is not advertised in the IP routing protocol is referred to as a *hidden bypass*. In contrast to the traditional bypass with NE, where dynamic circuits established in the transport network are used as regular new IP links, a hidden bypass offloads IP traffic onto the optical layer under the following conditions:

- The bypass must not be advertised as a link in the IP routing service.
- Only the router at the ingress of the bypass is configured to offload specific, predefined IP flows onto the bypass, while other routers remain unaffected by the bypass' existence.
- An IP flow can only be offloaded onto a bypass if the ingress

and egress of that bypass lays on the original IP routing path of that flow.

Under these conditions, we minimize the impact of bypass setup on the IP routing stability compared to the introduction of a conventional network engineered IP link. An example of such a hidden bypass is shown in Fig. 2.9 between routers b and e. Assuming shortest path routing and traffic from b to e would originally use the path $(b \rightarrow a \rightarrow e)$, this flow could be offloaded onto the bypass, since the bypass ingress router b and the bypass egress router e lay in the original routing path. A counter-example is the flow from b to d that could not be offloaded onto the bypass since the bypass egress router e does not lay in the original routing path $(b \rightarrow c \rightarrow d)$.

The possibility to establish hidden bypasses actually depends on the deployed hardware in the operator's network, since an IP router must be configurable in a way that it a) doesn't advertise certain new links via the routing protocol and b) can re-route traffic on a per flow basis. In order to distinguish the two Switch-On schemes in this thesis, we use the abbreviations *Switch-On BP* in case we use the hidden bypasses, whereas we say *Switch-On NE* in case we use conventional Network Engineering.

In our model, we assume that a *logical* IP link (or hidden bypass) is established by using a single or multiple (aggregated) *physical* IP links. There are no restrictions (in terms of numbers or size) of IP links that can be aggregated, e.g. a 10 Gbit/s and a 40 Gbit/s circuit can be aggregated in order to establish a 50 Gbit/s logical IP link. However, and without the loss of generality, we do not allow the virtualization of an IP port: for example a single 100 Gbit/s IP port cannot be used to provision two different links. In the link Switch-Off scheme, either all physical links constituting a logical IP link are switched off to shut down the connection completely or some of the physical links are shut down to reduce the capacity of a logical link. On the other hand, in the Switch-On scheme,

additional physical links can be activated to boost the capacity of an existing link or to establish a completely new logical IP link or hidden bypass. As mentioned before, we do not constrain the total number of ports installed on an IP router. Whereas our results will show that the *total capacity* installed in the Switch-On scheme is lower making this a fair assumption, we see in our performance evaluation that the optimal solution does require a larger number of small (low capacity) IP ports. Also note that in our formulation we assume a perfect matching between the capacity of the port and the corresponding circuit which means that a 10 Gbit/s circuit cannot be associated with a 40 Gbit/s IP port.

2.7 Energy Management

For a general and accurate power model, there are too many parameters to consider, e.g. technology, vendor, performance, generation and utilization of the network equipment, applications run in the network, etc, see [25], with many of them not easy to obtain. Therefore, in this thesis we use a simplified power model, which depends on two parameters only: 1) the number and type of active IP ports during a time interval and 2) power consumption of the circuits provisioned in the transport layer to establish IP links or hidden bypasses. The power consumption of an IP port is a (known) function of the port granularity. Note here that we assume a bandwidth discount, i.e., the power per bit decreases with increase in port sizes so that the power consumed by a 40 Gbit/s IP port would be lower than the power consumed by four 10 Gbit/s IP ports. As mentioned before, we assume that the WDM network is used to provision IP links and for a given pair of IP routers, all circuits use the same physical path. The power consumed by a circuit is defined as a function of the optical technology used and the number of physical hops in the optical layer. The technology used can be opaque or transparent optical

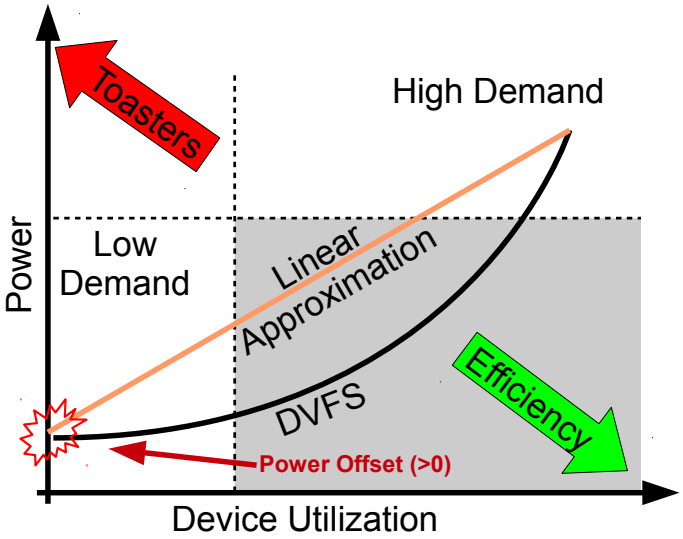


Figure 2.10: Power profiles for DVFS.

network, depending on the regeneration requirement of optical signals. In general, however, our power model is technology agnostic, since the power parameters used can be adapted to reflect conditions in any optical transport network technology, with or without optical signal regeneration. Finally, it is assumed that the IP router chassis used are the same in all cases, so that the chassis power consumption is not included in our model. This simplification is justified by our results which show that *total capacity* installed is lower in the Switch-On scheme ensuring that no unfair advantage is given to it.

Multiple approaches for energy conservation in networking equipment have been proposed, but only a few have been standardized and implemented. However, in this thesis we consider a type of energy saving mechanism that allows *gradual* power reduction relative to the utilization of the networking device. One mechanism that

Figure 2.11: Buffer thresholds can control energy management.

shows that kind of power profile is called Dynamic Voltage and Frequency Scaling. *Dynamic Frequency Scaling* is the technique where the clock frequency is reduced when the demand is low, which *proportionally* reduces the power consumption (and hence heat generation). Because the voltage for stable operation is depending on the clock frequency, the voltage can be reduced whenever the frequency is reduced. This technique is called *Dynamic Voltage Scaling*, which can *quadratically* reduce the power consumption (and heat). Using both principles together is referred to as Dynamic Voltage and Frequency Scaling (DVFS), a prevalent energy saving technique of CPUs for mobile computing. The mathematical relationship between the parameters is

$$P_{\text{dynamic}} = CfV^2 + P_{\text{static}}$$

Here, C is a fixed parameter for the chip's capacitance (that depends on the transistor count and transistor gates' structure size, i.e., the *feature size* of the chip). f is the clock frequency, V is the supply voltage, and P_{static} is some additional (but constant) basic power consumption. It is easy to see from the equation that DVFS is not linear due to the quadratic contribution of the voltage, and an approximated characteristic curve is shown in Figure 2.10. For complexity reasons, we assume a linear relationship between utilization (i.e., performance) and power consumption, depicted as the linear (orange) graph in Figure 2.10.

As it can be seen from state-of-the-art CPUs, today's CMOS technology allows for an increasing number of clock and voltage domains on the same chip. If we assume that future architectures for IP routers can make good use of these features, some functional blocks that have to be executed in line rate (e.g., the physical layer functional blocks at the ports, buffers, etc.) can be separated from the blocks dedicated for table lookup, packet processing, etc. In this way, the latter blocks can reduce their power under low traffic demand. We furthermore assume that the most functional units of the chassis can also be clocked down during low demand times. An intuitive approach for an energy manager would be to regulate the clock frequency of a block depending on the fill level of the buffer it is fed from. This is illustrated in Figure 2.11, where in case the ingress buffer of a port shows a very low buffer fill level, the associated functional blocks behind could be slowed down stepwise. When the buffer fill level reaches a certain threshold, the clock frequency is not decreased anymore, and by reaching another (higher) threshold, the clock frequency is increased stepwise. Please note that more sophisticated approaches for energy management have been proposed (e.g. dynamically reconfigurable modulation formats for flexible adjustments of the used transmission rates, etc.), but are not further considered in this thesis.

3
Planning of Hybrid Legacy/SDN Networks

3.1 Introduction

This chapter addresses the network administrative tasks that are related to the planning and engineering of IP networks that use a hybrid legacy/SDN control plane. The greenfield design of a network (i.e. its planning from scratch), is however not discussed here, as such an undertaking is known to be based on business considerations, e.g. future market situation, price trends, regulatory circumstances, forecasted customer quantity, etc., rather than on technological considerations. On the contrary, the here used notion of the term network planning considers only technical aspects and follows the definition of "offline planning" in [26]: *"Designing and reconfiguring a (virtual) network based on a long-term traffic scenario."* According to this notion, the planning of an IP network has the purpose of determining the dimensions and capabilities of its resources, such as link capacities and operational features of the routers.

The mathematical models of hybrid legacy/SDN networks for network planning tasks presented in this chapter are new and allow for the optimal network clustering for SDNp, optimal node placement for regular (i.e. non-SDNp) SDN/OSPF operation, link capacity dimensioning, and the calculation of the minimum provisions for fault tolerant operation.

3.2 Supporting Publications

1. M. Caria and A. Jukan, "On the IP traffic matrix problem in hybrid SDN/OSPF networks," *submitted for review to IEEE Transactions on Network and Service Management.* (Preprint available at arXiv.org: `https://arxiv.org/abs/1610.08256`)

2. M. Caria, A. Jukan, and M. Hoffmann, "SDN Partitioning: A centralized control plane for distributed routing protocols," *IEEE Transactions on Network and Service Management,* Volume 13, Number 3, pp. 381-393, September 2016. (Preprint available at arXiv.org: `https://arxiv.org/abs/1604.04634`)

3. M. Caria and A. Jukan, "Link capacity planning for fault tolerant operation in hybrid SDN/OSPF networks," *accepted for publication at GLOBECOM 2016*, Washington, USA, December 2016. (Preprint available at arXiv.org: `https://arxiv.org/abs/1604.05534`)

4. M. Caria, A. Jukan, and M. Hoffmann, "A performance study of network migration to SDN-enabled traffic engineering," in *Global Communications Conference (GLOBECOM), 2013 IEEE,* December 2013, pp. 1391-1396.

3.3 Network Clustering

The operational scheme SDNp requires a partitioning (or clustering, in networking parlance) of the routing domain into sub-domains, that are connected only by SDN-enabled border nodes. The network planning task of placing the SDN nodes thus entirely determines the size and number of sub-domains in SDNp-operated networks. Finding a good partitioning for a given topology is basically a graph theoretical discipline called graph partitioning, which has applications in many scientific and technical areas. Formally, the partitioning of

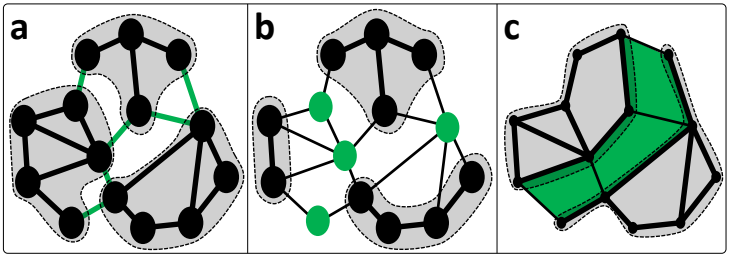


Figure 3.1: The three ways to partition a graph.

a graph $\mathcal{G} = (\mathcal{N}, \mathcal{L})$ is a grouping of the graph's nodes into K subgraphs $\mathcal{G}_1, \ldots, \mathcal{G}_K$, such that every node is included in one and only one of the K subgraphs. More precisely, we define a K-way partitioning of graph $\mathcal{G}$ as $\{\mathcal{G}_1, \ldots, \mathcal{G}_K\}$ with $\bigcup_i \mathcal{G}_i = \mathcal{N}$ and $\bigcap_i \mathcal{G}_i = \emptyset$. Graph partitioning is in general achieved by the (logical) deletion (or *cut*) of some type of graph element, like shown in Figure 3.1, either by removing a) edges, b) vertices, or c) faces. In our case, as the application of graph partitioning is the segmentation of an OSPF domain into nonadjacent sub-domains by determining a set of SDN-enabled sub-domain border nodes, the graph theoretical problem we are dealing with is the search for a so-called *vertex separator*. This is a set of nodes $\mathcal{G}_0 \subset \mathcal{N}$, such that the removal of $\mathcal{G}_0$ partitions $\mathcal{G}$ into K mutually unconnected subgraphs $\mathcal{G}_1, \ldots \mathcal{G}_K$. We assume that a graph we aim to partition is undirected, as we model IP networks, where all links are bidirectional.

3.3.1 ILP Model

For the most applications, determining a set of edges to be cut (the so called *cut set*) is the natural approach for graph partitioning, which is also why the vast majority of algorithms that can be found in the literature try to find a minimum cut set with some balancing

constraint regarding the number (or aggregated weight) of nodes in the subgraphs [27]. A straightforward approach to find a good vertex separator is therefore to adapt one of these algorithms, simply by letting it derive a minimum cut set and to find a minimum adjacent node set (i.e., a minimum set of nodes, whose removal also removes the cut set), which however is not necessarily optimal. We therefore formulated an ILP model of network partitioning, which is based on the vertex separator ILP model published by Balas and de Souza in [28]. While the latter is limited to only two subgraphs, constrains an upper bound on the imbalance of the subgraph sizes, and has the objective of minimizing the vertex separator (i.e., the number of SDN border nodes in our application), we find our approach more applicable to our specific networking problem: It allows for the partitioning of a graph into an arbitrary number K of subgraphs, constrains the number of SDN nodes, and has the objective to balance the sizes of all K subgraphs. Our approach results in extremely balanced subgraph sizes. The fundamental idea remains however the same: Each node either belongs to a single connected subgraph or to the vertex separator, thus we can demand that a node belongs to the same subgraph as any of its direct neighbors, unless the node or the neighbor is an SDN node. This way, we force all node subsets to be pairwise unconnected and disjunct. We then balance the partitioning by putting a punishment cost on each subgraph that quadratically increases with its size, and then minimize the total cost over all subgraphs. We used the notation shown in Table 3.1.

Our objective function is defined as

$$\text{Minimize} \sum_{k=1}^{K} \kappa(k) \tag{3.1}$$

This objective requires a constraint that determines the cost per sub-domain, such that it increases, as described above, quadratically with its size. However, as a quadratically increasing function can not

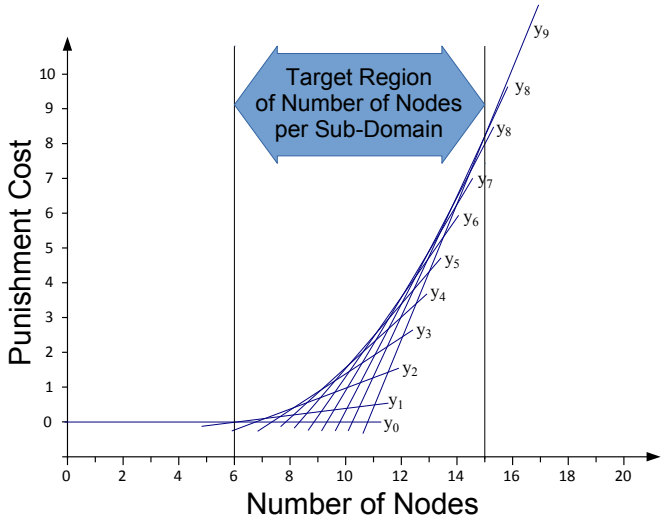


Figure 3.2: The linear cost functions for network clustering.

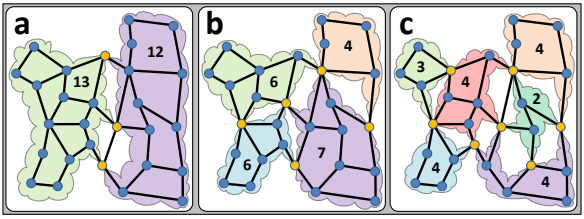


Figure 3.3: Nobel-EU topology partitioned into 2, 4, and 6 sub-domains.

Parameter	Type	Meaning
$\mathcal{N}$	Set	The nodes of the network
$\mathcal{L}$	Set	The (bidirectional) links of the network
MaxSDNnodes	Integer	Max number of SDN-enabled nodes
K	Integer	Number of subgraphs
$\mathcal{Y}$	Set	Set of linear functions that jointly resemble the lower bound of a quadratically increasing cost
Variable	**Type**	**Meaning**
$\kappa(k)$	Real	Cost assigned to subgraph k
$\varepsilon(k)$	Integer	Number of nodes in subgraph k
$\gamma(n,k)$	Boolean	Node n is part of subgraph k
$\mu(n)$	Boolean	Node n is a border node

Table 3.1: Variables for network clustering.

be used in linear optimization, we have to determine the cost increase with a piecewise linear inequation over a set Y of linear functions:

$$\forall y \in \mathcal{Y}, \forall k \in [1, K]: \quad \kappa(k) \geq y_a \cdot \varepsilon(k) + y_b \tag{3.2}$$

We here apply the linear cost functions $y \in \mathcal{Y}$ like shown in Figure 3.2, which allows to resemble a quadratically increasing punishment cost over a specific (pre-defined) target range of sub-domain sizes with linear functions. Each linear function is used as lower bound on the punishment cost. This method is similarly used in the traffic engineering model in Subsection 4.3 and provides better results than for instance the minimization of the maximum sub-domain size. The definition of the target range is not required to be very

precise. It is sufficient to roughly estimate by the following rule of thump: A network with $|\mathcal{N}|$ nodes that is to be partitioned into K sub-domains should not have more than $\frac{|\mathcal{N}|}{K}$ nodes per sub-domain, as a subset of the nodes will belong to the vertex separator (and thus not count to any of the sub-domains). However, some topologies may be difficult to partition such that all sub-domains exhibit a node count within the targeted range. The ILP model will nonetheless solve, but with a non-optimal clustering. In these cases, the target range can be redefined in width and position to adapt to the specific topology.

The following constraint determines the size $\varepsilon(k)$ of sub-domain k simply by counting all its nodes:

$$\forall k \in [1, K]: \quad \varepsilon(k) = \sum_{n \in \mathcal{N}} \gamma(n, k) \tag{3.3}$$

This in turn requires a constraint that assures that each node is either an SDN node or it belongs to a single sub-domain:

$$\forall n \in \mathcal{N}: \quad \mu(n) + \sum_{k=1}^{K} \gamma(n, k) = 1 \tag{3.4}$$

Neighboring nodes can not belong to different sub-domains:

$$\forall k \in [1, K],\ \forall i, j \in \mathcal{N} \ \text{ with } \ (i, j) \in \mathcal{L}: \qquad \gamma(i, k) + \mu(i) \geq \gamma(j, k) \tag{3.5}$$

The constraint assures (on the right-hand side with $\gamma(j, k)$) that node j belongs to sub-domain k only if its neighbor i either belongs to the same sub-domain (left-hand side: $\gamma(i, k)$) or is an SDN node (left-hand side: $\mu(i)$). Finally, the next constraint limits the total number of nodes that can be SDN-enabled to the upper bound $\mathsf{MaxSDNnodes}$:

$$\sum_{n \in \mathcal{N}} \mu(n) \leq \mathsf{MaxSDNnodes} \tag{3.6}$$

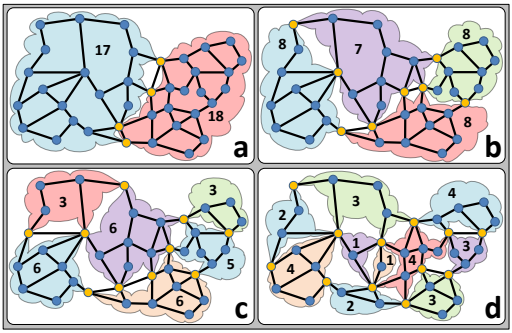


Figure 3.4: Janos-US-CA topology partitioned into 2, 4, 6, and 10 sub-domains.

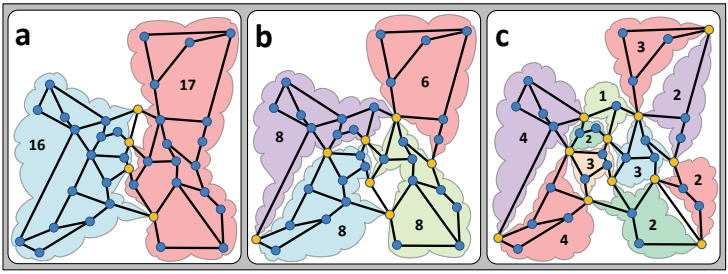


Figure 3.5: Cost266 topology partitioned into 2, 4, and 10 sub-domains.

The model solves quickly for networks with $|\mathcal{N}| < 50$, which allows to fine tune parameters K, MaxSDNnodes, and to additionally introduce lower and upper bounds $lb \leq \varepsilon \leq ub$ on the subgraph size. The partitioning of three example topologies is illustrated in Figures 3.5, 3.4, and 3.3. The shown topologies and their clustering is used in the following sections for the numerical evaluations.

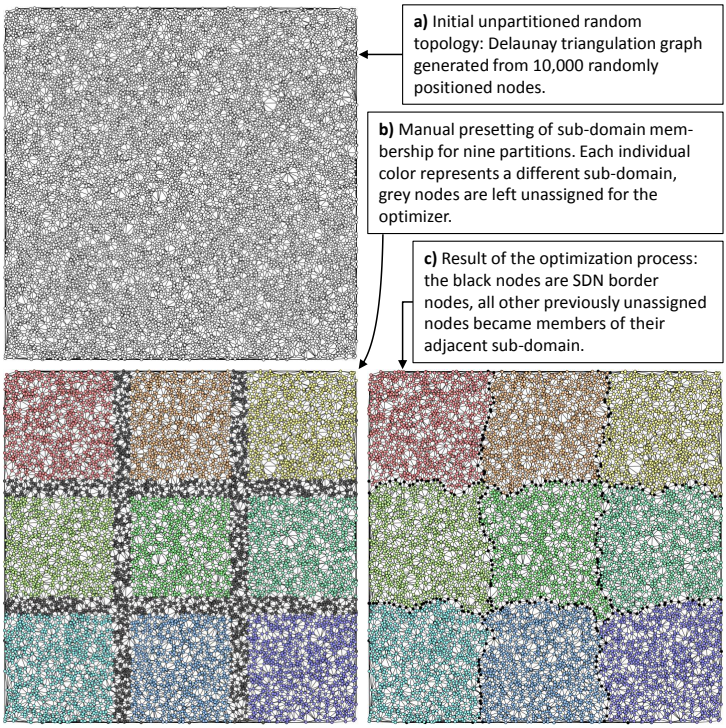


Figure 3.6: Partitioning of a 10,000 nodes random graph.

3.3.2 Reducing the Complexity for Large Topologies

In larger routing domains, e.g., with a number of nodes $|\mathcal{N}| >> 100$, the shown ILP model can not be regarded as a "turnkey" solution, due to its computational complexity. This problem can be worked around by manually pre-assigning large node areas to sub-domains. Figure 3.6 shows how manual pre-partitioning allows the usage of the

proposed ILP to partition a 10,000 node planar random topology[1]. As network operators usually have a topographical visualization of their topology (which they need for monitoring, network planning, debugging, and other network management tasks), large subsets of nodes can be assigned according to the preference of the operator. The example in Figure 3.6 shows a very simple pre-partitioning into 9 subgraphs (Figure 3.6b), leaving enough "meat" (i.e. free variables) for the solver to find a sufficient balancing of subgraph sizes. Indeed, the resulting clustering (shown in Figure 3.6c) required only 335 border nodes to split 10,000 nodes into 9 subgraphs, all with 1074 ± 2 nodes.

3.4 Node Placement

This section models the problem of SDN node placement in regular (i.e. non-SDNp) hybrid SDN/OSPF networks for two different objectives, which are a) SDN migration (which will be addressed in Subsection 3.4.1) and b) traffic monitoring (which will be addressed in Subsection 3.4.2). Both objectives are addressed individually before the compatibility of the two node deployment schemes are analyzed in Subsection 3.4.3.

3.4.1 SDN Node Placement for Migration Planning

A natural question for every network operator is the issue of migration from IP routers to SDN-enabled equipment, e.g., OpenFlow switches. On the one hand, not all routers in the network domain can migrate to SDN at once, due to operational and economic constraints. On the other hand, since SDN allows for more sophisticated traffic engineering in the packet layer compared to the common IP routing protocols with destination-based forwarding and least-cost

[1] The graph was generated by a Delaunay triangulation [29] of 10,000 randomly positioned nodes.

routing, the immediate benefits in traffic engineering are obvious. However, if we assume that a typical medium to large scale ISP will not migrate to SDN at once, but in a multi-period planning cycle (that may last for years), the network operator might want to know which network nodes should be migrated first, and which subsequently, etc. To this end, it is important to understand how a network can make best use of SDN-enabled traffic engineering during all stages of a migration process. Especially those stages are of interest which assume that the native IP routing, such as OSPF, coexists with SDN-enabled traffic engineered routing. Please note that our focus here is on the IP layer and not on network technologies with built-in traffic engineering capabilities (e.g. MPLS or Carrier Ethernet). However, there are ongoing efforts in standardization committees, industry, and academia to integrate these technologies into SDN.

This section details a two-stage algorithm that was presented in [30]. It optimizes the sequence of nodes for SDN migration, referred to as *migration scheduling*. In the first stage, the algorithm analyzes the network topology in order to find all path candidates for traffic engineering, and to identify routers (nodes) that have to be SDN migrated to allow the usage of the said paths. The second stage is the optimization of the scheduling with the objective to maximize the total number of path alternatives over the whole migration planning horizon. Our initial results show that an optimized migration scheduling can increase the number of alternative paths notably, especially in the early migration stages. This implies that just a few properly located SDN routers can provide a significant performance improvement. To evaluate how this result translates into capacity savings through traffic engineering, we also develop a corresponding performance benchmark. To this end, we first simulate traffic, including the growth of traffic over the migration planning horizon. After that, we optimize the routing in each migration period – ac-

cording to the available paths through the nodes already migrated in that period – with the objective to minimize the installed link capacities. The results obtained show that our approach is effective and can result in significant capacity savings up to 16% already in earliest stages of the migration time horizon, as compared to the 9% of an SDN migration with randomly chosen nodes for migration.

Since our main goal in this section is to find an optimal sequence of nodes to migrate to SDN, capacity planning in the lower layers is out of scope. We therefore refer to the comprehensive work on planning techniques for optical core networks [31], where one of the most important findings is that the network operator's choice of the prediction interval for the demand and cost forecast is important. By choosing a short prediction interval, it may be not possible to take the future evolutions sufficiently into account, while choosing a long prediction interval may suffer from uncertain predictions. An efficient approach using an ant colony meta heuristic for the multi-layer multi-period migration problem formulated as a path finding problem is proposed in [32]. Stochastic programming approaches like [33] have shown to better handle the uncertainties in multi-period network planning, when the future can be classified into only a few different scenarios.

As we model our migration scenario as a multi-period planning problem, the decision on how long the planning horizon should be (i.e., after how many years should the network be fully SDN-enabled), is important for the network operator. To this end, we refer to [34], where the timing issues of migration to a new technology were studied, and it was shown that demand growth, migration cost, and cost savings from the new technology have to be taken into consideration. In [35] the number and location of SDN controllers is studied, which is an important related aspect, but is outside the scope of this thesis.

As we consider a multi-period network migration (e.g., duration of five years, with ten 6-month migration periods), we take into ac-

count the growth of traffic load over the entire planning horizon. Without migration, an expectation is that link capacities have to be upgraded as the traffic grows; our goal here is to study whether advanced traffic engineering capabilities of SDN-enabled routers may be equally sufficient to reduce the amount of capacity upgrades. For instance, if there is 12 Gb/s traffic load on a certain link in the network, an upgrade from 10Gb/s to 40 Gb/s maybe necessary, which would require installation of 40 Gbit/s ports on both ends, while it might be possible to re-route a few traffic flows on the said link in case SDN routers are already deployed along that path.

The migration strategy that we propose here is based on a two stage heuristic algorithm to calculate an order of network nodes advantageous for the network operator to be used as migration sequence. An illustrative flow chart of our algorithm is shown in Figure 3.7: the ovals depict the input data and the rounded rectangles show the functions used. The two functions (i.e., stages) of the algorithm are topology analysis and migration sequence computation. The performance analysis ("benchmark") is illustrated in the lower part of Figure 3.7. This benchmark simulates for the given topology information and node sequence the traffic growth over the migration planning horizon and optimizes the network by means of combined capacity planning and traffic engineering for each migration period. As such, the performance benchmark is used to evaluate how the proposed migration strategy actually translates into capacity savings in the network. It can be seen that, for the benchmark, the available paths and the information on already migrated nodes are input from the scheduling algorithm (upper part). The benchmark starts in migration period 0, in which there is no router migrated yet, which means that all flows are routed via their least-cost paths. In this way, the network optimization is restricted to just choose the smallest possible link capacities according to the flows they carry. Then, the scenario changes to the next migration period (depicted

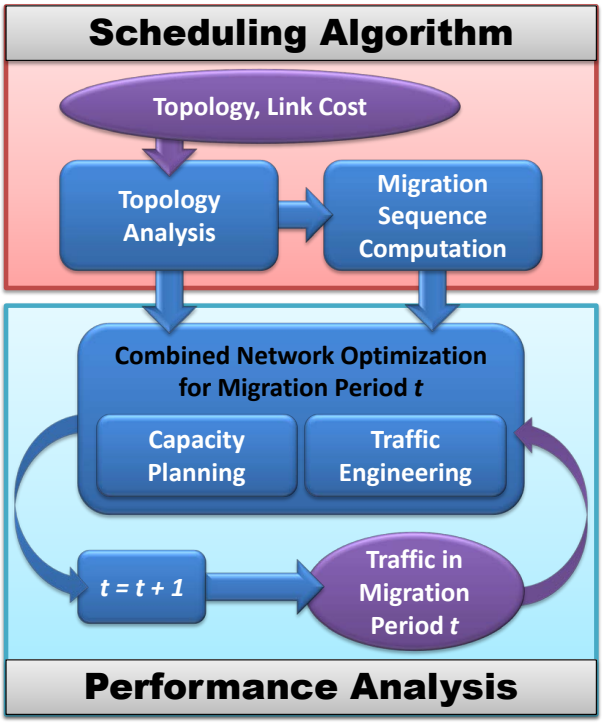


Figure 3.7: The proposed algorithm and performance benchmark.

by the "t=t+1" box) and the traffic load on all flows is increased, so that a combined network optimization can be performed with the new set of parameters. This analysis is iterated over all migration periods, so that the final result of our migration scheduling can be evaluated by comparing it to a random migration sequence. The two migration sequences (optimized vs. random) are compared based on how much the capacity upgrades (which are necessary due to the growing traffic) can be reduced over time.

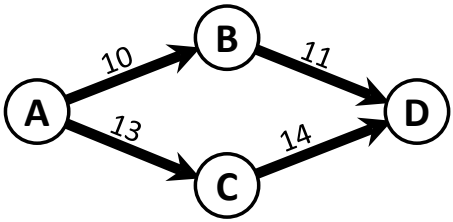
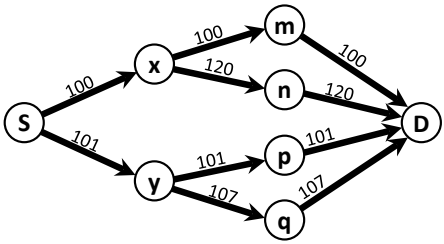


Figure 3.8: The maximum difference between link costs.

The first step of our algorithm is to compute all shortest paths in the topology. To reduce the problem complexity, we here only consider paths of the same length (i.e., same hop count) compared to the least-cost OSPF routes as path alternatives for traffic engineering. While SDN-capable routers could actually route flows on longer paths, we justify this restriction with the facts that 1) longer paths (i.e., more hops) imply a reduced Quality of Service for the users due to an increased packet delay, and 2) flows routed over more links than necessary waste more capacity installed. However, please note that there are possible cases in which re-routing flows to longer paths could save capacity, e.g., when re-routing a flow from a poorly utilized 40 Gbit/s link to a longer path with enough spare capacity, so that a 10 Gbit/s link could be used instead. To assure shortest path routing, we assume that the difference between any two link costs is below the inverse of the global maximum path length relative to the minimum link cost in the topology. In other words, this constraint assures that no path with n hops between any node pair can have a higher path cost than any path with more than n hops between the same node pair. In fact, the constraint provokes that only shortest paths can be used for routing, as it assures that more hops must result in a higher link cost in any case. Please note that leaving out this assumption would only result in slightly improved SDN traffic engineering capabilities. Figure 3.8



Path	(S→x→m→D)	(S→x→n→D)	(S→y→p→D)	(S→y→q→D)
Key Nodes	{}	{x}	{S}	{S, y}

Figure 3.9: Key nodes.

illustrates the idea: The maximum path length is $L_{\max} = 2$ (i.e., $A \rightarrow B \rightarrow D$ and $A \rightarrow C \rightarrow D$), and the minimum link cost is $W_{\min} = W(A \rightarrow B) = 10$. Hence, the maximum allowed difference between any two link costs is $\Delta_{\max} < W_{\min}/L_{\max} = 10/2 = 5$. It can be seen in Figure 3.8 that the shown link costs meet the condition: $\Delta_{\max} = 14 - 10 = 4 < 5$.

The second step of the topology analysis is to identify the so-called *key nodes* on all paths. We define a key node n on path p as a node which has to be SDN-migrated in order to allow the usage of path p for traffic engineering. The fact that certain paths can not be used for traffic engineering without SDN capabilities is due to the nature of destination-based forwarding and least-cost routing. Figure 3.9 illustrates the context: There are four different paths from S to D, via node m, n, p, or q. Without any SDN migrated node, only the m-path can be used, because it is the least-cost. That means node S will forward any traffic for D towards node x without taking into account what x will do with that traffic. The reason for that is that S learned through the routing protocol from x that it can reach D via x with an accumulated cost of 300, which is minimum for S. In case node x is already migrated to SDN, the path via node n

can also be assigned for the traffic between S and D by the traffic engineering algorithm, because x can be configured by the central SDN controller to forward traffic with source S and destination D to node n. This example shows why x (and only x) is called the key node on the path from S to D via n, and how SDN eliminates the least-cost routing and destination based forwarding constraints. Accordingly, the p-path has node S, and the q-path has nodes S and y as key nodes.

However, ordering the nodes in the migration sequence according to the number of paths in which they appear as a key node is not necessarily leading to a good solution (i.e., as many paths as possible as early as possible), as it can be seen from the following example: Assume five arbitrary paths, here identified only by the set of their key nodes: $\{a\}$, $\{a, b, x\}$, $\{b, c, x\}$, $\{c, d, x\}$, and $\{d\}$. Obviously, x appears in the most paths as key node, but in all four optimal migration sequences *adxbc*, *adxcb*, *daxbc*, and *daxcb*, the node x appears as third one. Of course, the results can worsen further in more complex scenarios, which is why we developed the mathematical model described below.

ILP Model

We now present the ILP model we used to determine the schedule (i.e., the sequence of nodes) in a multi-period migration scenario. The used notation is shown in Table 3.2. We model all migration periods including τ_0, which represents the technological state of the network before migration, i.e., with zero nodes migrated. In the final period $\tau = \tau_{\mathrm{T}}$, all nodes have been migrated. This means that independently of the migration sequence, all pre-calculated paths $p \in \mathcal{P}$ (which is the input coming from the topology analysis) are available for traffic engineering. In the previous subsection, we defined a key node n of a path p as a node that must be migrated in order to allow the usage of p for traffic engineering. This means that p is in fact not

Parameter	Meaning
T	Number of migration periods
τ_i	The i^{th} period of the migration
τ_1, τ_{T}	The first and the last period of the migration
τ_0	The time period representing the state before migration
$\mathsf{Priority}(p)$	Priority of path p
$\mathsf{KeyNodes}(p)$	Number of key nodes on path p
$\mathsf{isKey}(n, p)$	Boolean parameter, true if node n is a key node on path p
Variable	**Meaning**
$\eta(n, \tau)$	Boolean, true if node n is already migrated in period τ
$\pi(p, \tau)$	Boolean, true if path p is already available in period τ

Table 3.2: Notation for SDN migration planning.

the least cost route. Therefore, the availability $\pi(p, \tau)$ of a path p in period τ depends on whether all its key nodes have already migrated in that period. The following constraint models this feature:

$$\begin{gathered} \forall p \in \mathcal{P},\ \forall \tau \in \{\tau_0 \dots \tau_{\text{T}}\} : \\ \pi(p, \tau) \cdot \mathsf{KeyNodes}(p) \leq \sum_{n \in \mathcal{N}} \mathsf{isKey}(n, p) \cdot \eta(n, \tau) \end{gathered} \tag{3.7}$$

The number of nodes that can migrate per migration period has to be limited to the number of nodes divided by the total number of

migration periods. We model this with the following constraint:

$$\forall \tau \in \{\tau_0 \dots \tau_{\mathrm{T}}\} : \qquad \sum_{n \in \mathcal{N}} \eta(n, \tau) \leq ind(\tau) \cdot \left\lceil \frac{|\mathcal{N}|}{\mathrm{T}} \right\rceil \tag{3.8}$$

In case $|\mathcal{N}|$ can't be divided by T without a remainder, we simply round up, which leads to a final migration period with a lower number of nodes that migrate. Finally, we also have to constrain the trivial fact that we disallow any node to "migrate backwards":

$$\forall n \in \mathcal{N},\ \forall \tau_{\mathrm{t}} \in \{\tau_0 \dots \tau_{\mathrm{T\text{-}1}}\} : \qquad \eta(n, \tau_{\mathrm{t}}) \leq \eta(n, \tau_{\mathrm{t}+1}) \tag{3.9}$$

The objective function is to maximize the number of paths used for traffic engineering, summarized over the whole planning horizon, i.e.,

$$\text{Maximize} \quad \sum_{t=1}^{\mathrm{T}} \sum_{p \in \mathcal{P}} \pi(p, \tau_t) \cdot \mathsf{Priority}(p) \tag{3.10}$$

We leave the question how a path is used for traffic engineering (i.e., what values are assigned to all $\mathsf{Priority}(p)$) to be answered individually by the user of our model, as we do not provide a general answer. We assume that an optimal migration scheduling can not be found by only determining $\mathsf{Priority}(p)$, because for an optimal migration scheduling one would definitely have to take the process of traffic engineering during all migration periods into consideration. For illustration, we define three different strategies to determine $\mathsf{Priority}(p)$, i.e., three different heuristics to find migration schedules:

– **Number of paths:** In this strategy, the value $\mathsf{Priority}(p)$ of each path p is set to 1, so that all paths are assumed to be identically beneficial for traffic engineering. This leads to a migration schedule where nodes are migrated first, which provide the highest total number of path alternatives for traffic engineering.

– **Traffic:** This strategy sets the value $\mathsf{Priority}(p)$ of each path p to the initial traffic demand between its end nodes. This way, nodes

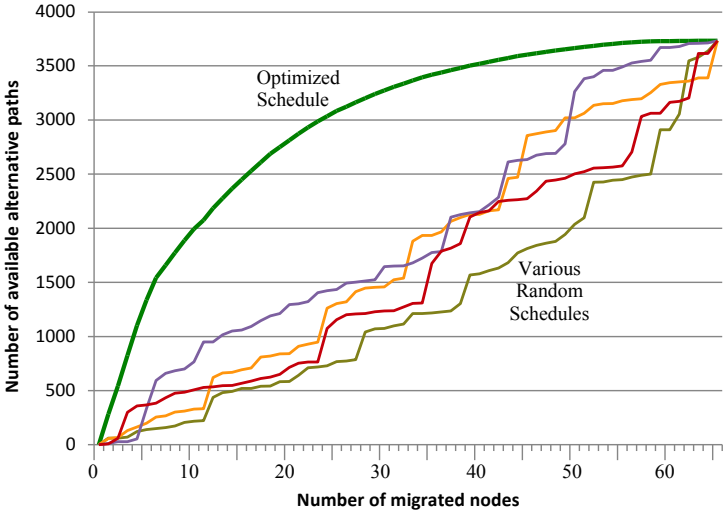


Figure 3.10: Comparison of optimized and random migration schedule.

are migrated first, which can provide the most path alternatives for high traffic flows.

– **Path diversity:** This strategy is a modification of the first one, having the same initial assignment of $\mathsf{Priority}(p) = 1$ for all paths. Then, for each s-d node pair, one path p (that is not the least cost path) is randomly chosen to be set to a higher $\mathsf{Priority}(p)$. This strategy leads to a migration schedule where already in the early migration at least one path alternative for as many existing least-cost routes as possible are available for TE.

Performance Evaluation

For our performance analysis we used the TA2 network from the SNDlib topology library [36], which has 65 nodes, 108 links, and

4160 traffic flows. The migration planning horizon was set to ten migration periods, in which routers had to be migrated. This time is partitioned into nine periods each with seven routers to migrate and a final period with only two routers. We have assigned the link weights randomly and averaged the results of the performance benchmark over ten different random assignments of link weights. The migration sequence optimization was computed in less than ten seconds, while the performance benchmark's traffic engineering part took up to 30 minutes (with an allowed MIP gap of 1%). For the performance benchmark we assumed that links are available in the following granularity: 1 Gbit/s, 5 Gbit/s, 10 Gbit/s, 40 Gbit/s, 100 Gbit/s, 400 Gbit/s, 1 Tbit/s. The traffic matrix was also randomly generated and the values are uniform distributed between 0 and 400 Mbit/s per traffic flow. For the traffic growth during the migration planning horizon we set the growth factor of each flow in each period to a random value between 1.05 and 1.35 so that the mean growth is 20% in every period. For every one of the ten sets of random link weights, we also generated an individual traffic scenario.

Figure 3.10 shows our initial result, which is the number of path alternatives, i.e., the paths that the TE algorithm can use to minimize the network capacity, depending on the number of migrated nodes. The green (monotonic) graph shows the result for an optimized migration schedule, while the other (zigzag) graphs show results for various random sequences of node migration. As it can be seen from the figure, all graphs start at the same point and end at the same point, which is due to the following fact. Before the migration process, none of the routers have migrated, so that there are also zero alternative paths available. All graphs also end at the same point, because independently of the migration schedule, all alternative paths are available after all nodes migrate to SDN. The most important insight from this initial result is that an optimized migration sequence shows a drastically increased number of alterna-

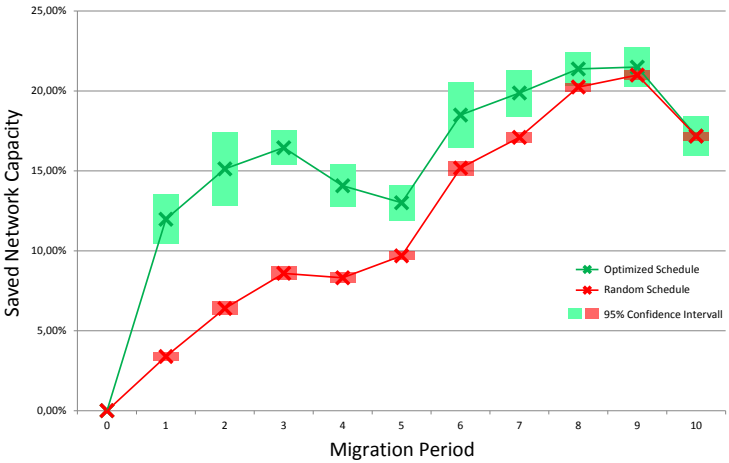


Figure 3.11: Benchmark result of optimized and random migration schedule.

tive paths through SDN nodes, especially during the early migration periods. The more nodes are migrated, the less is the difference compared to any random migration sequence.

We here evaluate how beneficial this increased number of alternative paths actually is for the network operator. Figure 3.11 shows the saved network capacity through traffic engineering in each migration period. The green graph is the result for an optimized migration order using the first heuristic that maximizes the total number of alternative paths. While with any migration sequence, SDN allows to save considerable amounts of network capacity through the alternative paths, it can also be seen from the figure that especially in the first third of the migration planning horizon (i.e., periods 1, 2, and 3), an optimized migration sequence clearly outperforms random sequences. The advantage of the optimal sequence is almost constantly at 8 percentage points over the random sequences during

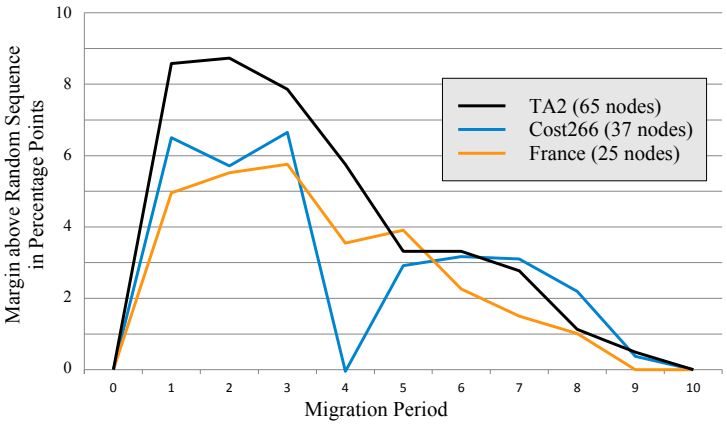


Figure 3.12: Optimized vs. random migration sequence.

this first third. An eye-catching point of the result is the performance drop between the fourth and the fifth period, and again at the tenth migration period. An analysis of the used port sizes in the result data showed that starting with the fourth period, multiple 400 Gbit/s links had to be used, which are likely to be underutilized in that early stage. Something similar happened in the final migration period, where for the first time multiple 1 Tbit/s links had to be used. In another study (not shown here), we also generated results for the other two migration sequence heuristics (i.e., the strategy that prefers alternative paths for high bandwidth flows, and the strategy that prefers alternative paths for flows which have no such alternatives yet). However, all three strategies exhibited the same performance (with differences within the confidence interval), and are therefore not shown here.

Finally, we evaluated the behavior of our scheme with two sample topologies, i.e., the Cost266 network (37 nodes, 57 links, 1332 traffic flows) as a medium size topology and the France network (25 nodes,

45 links, 600 traffic flows) as a small size topology. This allowed us to compare the outcome with the TA2 network (65 nodes, 108 links, 4160 traffic flows), already used in the previous subsection. In all three networks we set the number of migration periods to ten and the average traffic increase to 1.2 per migration period. However, in order to make the results comparable, we had to scale up the initial traffic for France and Cost266 networks, so that they show the same performance drop in the middle and at the end of the planning horizon, like seen in the TA2 network in Figure 3.11. The Cost266 network was set to 0–1.2 Gbit/s per flow and the France network was set to 0–2.5 Gbit/s per flow. Figure 3.12 shows only the margin between the optimized and the random migration sequence in percentage points. As it can be seen from the result, all topologies show comparable graphs, with a high advantage of the optimum over the random sequence in the first third of the migration planning horizon.

3.4.2 SDN Node Placement for Traffic Monitoring

Contrary to the previous subsection, we here determine the optimal SDN node locations for IP traffic monitoring, which is an important network management task. The IP traffic matrix determines the amount of traffic transferred per second between any ingress-egress pair of routers. It is essential for IP network operation and management, including tasks like traffic engineering, routing protocol configuration, security and reliability, capacity planning, and fault diagnosis. The traffic matrix is however not readily available in legacy IP networks. The measuring of all flows directly is not practical, as it requires a significant amount of monitoring equipment and network-wide configuration efforts [37]. Therefore, the traffic matrix is usually *estimated* or *sampled*, both leading to inaccurate results that may adversely impact network operations, due to faulty configurations based on uncertain traffic statistics.

Throughput statistics on a per-link basis (commonly referred to as *link loads*) are easily available from all network nodes via Simple Network Management Protocol (SNMP) requests and are typically monitored by network operators. The mathematical relation of the three parameters link load, routing, and traffic matrix is $L = R \cdot F$, where the link load of the n links in the network is the (given) column vector $L = (l_1, l_2, \ldots, l_n)^T$, the demand of all m ingress-egress (IE) flows is the (sought) row vector $F = (f_1, f_2, \ldots, f_m)$, and the routing is represented by the (given) binary $n \times m$ matrix R, where r_{ij} is 1 if flow j is routed via link i, and 0 otherwise. To put it more intuitively, the load on a link is the sum of all the IE flows that traverse it. Due to the fact that link loads represent *aggregated* flows with their number being (depending on the topology) easily an order of magnitude below the number of IE flows, an attempt to solve the above linear system for F results in a heavily under-determined linear system. The *estimation* of the traffic matrix based on this statistical data means the search for a *good* solution of the described problem,

which however typically exhibits severe estimation errors [38].

Traditionally, network operators used to deal with the problem either by significant over-provisioning of network resources (which renders the exact knowledge of the traffic matrix unnecessary), or by installing expensive monitoring equipment. However, the advent of Software-Defined Networking (SDN) involves a new and powerful mechanism for network monitoring, as SDN-enabled devices provide additional byte counters for all individual entries in their forwarding tables. We assume that this mechanism will solve the traffic matrix problem once and for all in the long run. Today, however, IP networks are likely to be implemented with a hybrid control plane deploying a distributed routing protocol like OSPF and the centralized paradigm of SDN in parallel. In hybrid networks, the known and difficult problem to generate the IP traffic matrix persists, as the required SDN byte counters may not be sufficiently implemented (or even not at all [39]), or the deployed SDN nodes may be too few, or not adequately located in the network.

This subsection addresses the network planning aspect of a new approach to solve the IP traffic matrix and the related monitoring problem in networks with a hybrid SDN/OSPF control plane. The basic functioning of this monitoring scheme is to augment the SDN-based traffic statistics with SNMP-based throughput measurements on IP backup links. The exact mode of operation, implementation aspects of the required hybrid monitoring infrastructure, and possible timing issues of the measurements are however detailed in Section 4.6 in the following chapter. This section deals only with the network planning aspect of the scheme, which is finding the optimal SDN node locations for traffic measurements, determining the backup links to deploy for additional measurements, and determine where to measure which traffic flow. We show in the performance evaluation that there is a near linear trade-off between both resources. We conclude from our results that a hybrid control plane

with only a few SDN nodes can provide the complete traffic matrix in case multiple backup links are available for measurements.

The problem of choosing the best nodes to perform flow monitoring has already been studied in the context of Cisco's IOS NetFlow feature. This traffic sampling method has impact on the CPU load in the router, which can be significant [40], and NetFlow must be available on the routers. This is not generally the case, especially in carrier networks, where multiple vendors' equipment is used. The authors of [41] point out that the accuracy of traffic analysis based on flow measurements depend on the sampling rate and the number and placement of monitors, and present methods to jointly optimize the problem.

A straightforward approach to measure the entire traffic matrix is to monitor traffic on all ingress routers' monitoring ports with cheap off-the-shelf host. A central server can then collect the data along with the routing information from all routers. This solution does however not scale to current transmission speeds in core networks with the typical 100 Gbit/s ports. There are systems on the market that do scale to core network dimensions, like HP's OpenView Dynamic Netvalue Analyzer [42]. However, such systems have to be purchased and maintained, and due to their involved high capital and operational expenditures, over-provisioning of network capacity till the point where having exact knowledge of the traffic matrix becomes unnecessary is still considered as the easiest and most cost effective solution by the majority of network operators.

In the case OpenFlow routers are deployed in the IP network, it is indeed possible to measure an IE flow directly on the ingress router. The network operator can identify all flow table entries for a specific egress router at the ingress router and use the according byte counters for this monitoring purpose. However, it has been argued in [39] that the required throughput statistics on a per-flow basis may be implemented insufficiently or even not at all.

Parameter	Meaning
$\mathcal{N}$	The set of all nodes in the network
$\mathcal{L}$	The set of all bidirectional links in the network
$\mathcal{L}'$	The set of all directional links in the network
$rev(\ell) = r$	The link reversion function for all $\ell \in \mathcal{L}'$, returns the reverse link r of link ℓ
$\mathcal{F}$	The set of all traffic flows in the network
$\mathcal{R}_x^f$	Boolean routing parameter, true if flow f traverses some resource $x \in (\mathcal{N} \cup \mathcal{L} \cup \mathcal{L}')$
$ind(x_i) = i$	The index number function, returns the i of x_i
$\mathsf{MaxFlows}(x)$	The maximum number of measurable flows on x
$\mathsf{MaxLoad}(x)$	The maximum measurable load on x
$cost(x)$	Cost for using resource x for measurements
$\sigma_{\min}$	Required min. number of flow determinations
Variable	**Meaning**
σ_x	Number of flow determinations due to the deployment of x
$\vartheta(x)$	Boolean, true if device x is provisioned
$\nu(f, x)$	Boolean, true if flow f is measured on device x
$\tilde{\nu}(f, \ell)$	Boolean, true if f is derived on ℓ from other measurements

Table 3.3: Notation for measurement location optimization.

ILP Model and Heuristic

In order to select the most beneficial backup links and nodes for SDN deployment, and to determine which flow is measured on which device (i.e. on which backup link or on which OpenFlow router), we developed two algorithms: a) an ILP-based, and b) a greedy heuristic. We here presents the related optimization model and algorithm in detail. A summary of the used notation is given in Table 3.3.

The objective function of the ILP model minimizes the total cost for the deployment of backup links and OpenFlow routers in the network:

$$\text{Minimize} \quad \sum_{x \in \mathcal{L} \cup \mathcal{N}} \vartheta(x) \cdot cost(x) \tag{3.11}$$

The duration of a backup link measurement cycle (i.e., the time for retrieving the necessary SNMP link counters for all desired measurements on a particular backup link) increases with the number of sequential measurements. It is however advantageous to limit all SNMP-based measurement cycles to the duration of a predefined global monitoring interval, which allows for synchronized and complete traffic snapshots at fixed instants in time. The following constraint allows such a limitation, and it additionally limits the number of flow table entries of an OpenFlow router that can be monitored with byte counters, which is practically relevant in case the router hardware is limited in this regard.

$$\forall x \in \mathcal{L} \cup \mathcal{N}: \quad \sum_{f \in \mathcal{F}} \nu(f, x) \leq \vartheta(x) \cdot \mathsf{MaxFlows}(x) \tag{3.12}$$

The right-hand side of the constraint furthermore sets the maximum number of measurable flows to zero in case the backup link or OpenFlow switch is not deployed. In the same way, we limit the traffic load on a certain measurement device (backup link or OpenFlow switch), which may be required for one of the following reasons: 1) An OpenFlow router's byte counter on flow table entries may be

implemented only in software, so that this function is not usable at full line rate. 2) The size of a backup link may be smaller than the size of the original link (e.g., its just a single port out of an Ethernet Link Aggregation Group), which can lead to an overutilization of the backup link when an elephant flow is measured. The following constraint can be used in such circumstances:

$$\forall n \in \mathcal{N}: \quad \sum_{f \in \mathcal{F}} \nu(f,n) \cdot \hat{f} \leq \vartheta(n) \cdot \mathsf{MaxLoad}(n) \tag{3.13}$$

The next constraint assures that every flow is either measured or derived from other measurements:

$$\forall f \in \mathcal{F}: \quad \sum_{\ell \in \mathcal{L}} \Big(\nu(f,\ell) + \tilde{\nu}(f,\ell) \Big) + \sum_{n \in \mathcal{N}} \nu(f,n) = 1 \tag{3.14}$$

If a particular flow is the only one left undetermined on a particular link, that flow can be calculated as the total link load minus the sum of all other flows' sizes on that link. The variable $\tilde{\nu}(f,\ell)$ is set to one in case the solver decides to derive a flow f on link ℓ in this way. However, to assure that on each link at max one flow is derived in this way, we need the following constraint:

$$\forall \ell \in \mathcal{L}: \quad \sum_{f \in \mathcal{F}} \tilde{\nu}(f,\ell) \leq 1 \tag{3.15}$$

Apparently, a flow can only be measured on a device if it's routing path traverse the device. This is taken care of by

$$\forall f \in \mathcal{F}, \ \ \forall x \in \mathcal{L} \cup \mathcal{N}: \quad \nu(f,x) \leq \mathcal{R}_x^f \tag{3.16}$$

$$\forall f \in \mathcal{F}, \ \ \forall \ell \in \mathcal{L}: \quad \tilde{\nu}(f,\ell) \leq \mathcal{R}_\ell^f \tag{3.17}$$

As we consider directional links in this model, whereas IP links are bidirectional, we additionally constrain backup links to be bidirectional. In other words, if link ℓ is provisioned with a backup, we require that its reverse link $rev(\ell)$ (i.e. the one that connects the

Input: Sets $\mathcal{N}$, $\mathcal{L}$, $\mathcal{F}$, Functions $\mathcal{R}_x^f$, $rev(\ell)$, Parameter $\sigma_{\min}$
Output: Set Ω of all required network resources

// Step 1: Initialization
$\Omega \leftarrow \emptyset$; $W \leftarrow \emptyset$; $R \leftarrow \mathcal{N} \cup \mathcal{L}$
foreach $w \in R$ **do**
  $\vec{w} \leftarrow \vec{0}$
  foreach $f \in \mathcal{F}$ **do**
    $i \leftarrow ind(f)$
    $\vec{w_i} \leftarrow \mathcal{R}_x^f$
  end
  $W \leftarrow W \cup \{\vec{w}\}$
end
while $|\bigvee W| > \sigma_{min}$ **do**
  // Step 2: Determine all W_x and σ_x
  foreach $x \in R$ **do**
    $W_x \leftarrow \emptyset$
    if $x \in \mathcal{L}$ **then** Link $r \leftarrow rev(x)$
    foreach $\vec{w} \in W \setminus \{\vec{x}, \vec{r}\}$ **do**
      if $x \in \mathcal{N}$ **then** $W_x \leftarrow W_x \cup \{\vec{w} \wedge \neg\vec{x}\}$
      else $W_x \leftarrow W_x \cup \{(\vec{w} \wedge \neg\vec{x}) \wedge \neg\vec{r}\}$
    end
    // Iteratively remove all calculable flows
    while $\exists \vec{\ell} \in W_x : |\vec{\ell}| = 1$ **do**
      $W_x \leftarrow W_x \setminus \{\vec{\ell}\}$
      foreach $\vec{w} \in W_x$ **do** $\vec{w} \leftarrow \vec{w} \wedge \neg\vec{\ell}$
    end
    $\sigma_x \leftarrow |\bigvee W| - |\bigvee W_x|$
  end
  // Step 3: Choose next resource x
  $x \leftarrow \arg\max_{z \in R}\{\sigma_z\}$
  $\Omega \leftarrow \Omega \cup \{x\}$; $R \leftarrow R \setminus \{x\}$; $W \leftarrow W_x$
end

Algorithm 2: Resource Provisioning for Traffic Monitoring

same nodes in the reverse direction) is provisioned with a backup too:

$$\forall \ell \in \mathcal{L}: \quad \vartheta(\ell) = \vartheta(rev(\ell)) \tag{3.18}$$

In addition to the ILP model discussed above, we provide here a greedy algorithm with its pseudocode shown in Algorithm 2, that can also be used as pre-stage to the ILP. This algorithm exhibits a time complexity which is orders of magnitude lower than the ILP and thus fast enough to provide solutions for large scale topologies. Its basic functioning is as follows: It determines in each iteration the next (yet not deployed) measurement resource (i.e. backup link or SDN node), whose deployment results in the largest number of determinable flows (that are yet undetermined). It therefore keeps track of which flows have been determined in previous iterations. This is necessary to avoid the case in which, for instance, there are two resources a and b, both providing the measurement of a very high number of flows, but the majority of the flows on a are the same as on b. A trivial greedy approach would simply choose both, while one of them would actually be redundant and without much benefit.

The heuristic algorithm uses a working set W of binary vectors (see Table 3.4 for the here introduced vector notation), which, in its initial state, represent the routing configuration in the network. Assuming that the network resources as well as the traffic flows of a network have a unique order, each vector $\vec{w} \in W$ represents a specific network resource (i.e. a node, or a directional link), and the i^{th} element of each vector represents the same i^{th} traffic flow. An element of a vector is 1 if the according flow traverses the according network resource, and 0 otherwise (see lines 5...8 in Algorithm 2). The objective of the algorithm is to identify in each iteration (of the *while*-loop in lines 11...30) the network resource, whose deployment would result in the maximum number of new flow determinations and deletes all corresponding flows from the vectors in the working

Vector	Meaning
$\vec{a}$	Undetermined flows on network resource a
$\vec{a_i}$	The i^{th} element of $\vec{a}$
$\vec{0}$	The zero vector $(0, 0, \ldots, 0)$
Connective	**Meaning**
$\neg\vec{a}$	The element-wise negation of $\vec{a}$
$\vec{a} \wedge \vec{b}$	The element-wise AND of $\vec{a}$ and $\vec{b}$
$\vec{a} \vee \vec{b}$	The element-wise OR of $\vec{a}$ and $\vec{b}$
$\vec{a} \leftarrow \vec{b}$	$\vec{a}$ is defined to be $\vec{b}$
$\lvert\vec{a}\rvert$	The cardinality (i.e. number of 1-bits) of $\vec{a}$
$\bigvee A$	The element-wise OR of all $\vec{a} \in A$

Table 3.4: Binary vector notation.

set W. Each iteration accordingly determines a single measurement resource (and stores it in the result set Ω, see line 29) in the following fashion.

The number of flow determinations σ_x due to the deployment of a specific resource (i.e. SDN node or backup link) x is calculated (in lines 13. . . 26) as follows: We define an empty test set of vectors W_x (line 14), and for each vector $\vec{w} \in W$ we add a vector $\vec{w'} \leftarrow \vec{w} \wedge \neg\vec{x}$ to W_x. Thus, W_x represents the routing of undetermined flows after resource x is deployed. Calculating σ_x for a backup link in parallel to a link x must take into account that backup links are bidirectional. In other words, the deployment of a backup link x implicates the deployment of another directional link $r = \text{rev}(\ell)$ in the opposite direction. This particularity is taken care of in line 18 (which is then executed instead of line 17), where for each vector $\vec{w} \in W$ we

add a vector $\vec{w'} \leftarrow (\vec{w} \wedge \neg\vec{x}) \wedge \neg\vec{r}$ to the set W_x.

Taking into account the *calculable* flows[2] due to a resource deployment is an iterative process, because the determination of the only remaining flow on a particular link always leads on all other links (that are traversed by the said flow) to have their number of undetermined flows decreased by one. This could in turn result in another link having a single flow undetermined. We therefore iterate (lines 21...24) over all link vectors $\vec{\ell} \in W_x$ with $|\vec{\ell}| = 1$ and delete the vector from W_x and then the according flow from all other vectors: $\forall\vec{w} \in W_x : \vec{w} \leftarrow \vec{w} \wedge \neg\vec{\ell}$, until there is no more such vector $\vec{\ell} \in W_x$ with $|\vec{\ell}| = 1$. We can finally calculate the number of flow determinations (line 25) as $\sigma_x \leftarrow |\vec{z}| - |\vec{z_x}|$, where $\vec{z} \leftarrow \bigvee W$ and $\vec{z_x} \leftarrow \bigvee W_x$.

The final step of each iteration (line 28) is to choose the resource x with the largest σ_x, and to remove all corresponding flows from the working set W for the next iteration (i.e. substituting W with W_x, line 29). The algorithm terminates when the number of remaining flows $\sigma = |\bigvee W|$ falls below a predefined threshold $\sigma_{\min} \geq 0$ (i.e. the break condition in line 11).

Please note that we here assume the generation of the complete traffic matrix from measurements and do not take into consideration any estimation method required when the traffic matrix is not complete. However, both proposed deployment strategies in this subsection, i.e. the ILP-based and the heuristic, can terminate with an incomplete traffic matrix, which would require a subsequent estimation of the remaining flows. We therefore refer to the flow spread metric that was proposed in [43] and represents the difference of the upper and lower bound of a flow, and thus provides a measure of *urgency* for the exact determination of the flow. This flow spread

[2] A flow is calculable, if that flow is the only one left undetermined on any (directional) link. Its throughput can then be derived by subtracting all of the known flows on that link from its link load. This is further explained in Section 4.6.

Topology	Nodes	Links	IE Flows	Degree
TA2	65	108	4160	3.32
Germany50	50	88	2450	3.52
Janos-US-CA	39	61	1482	3.13
Cost266	37	57	1332	3.08
India	35	80	1190	4.57
Nobel-EU	28	41	756	2.93
France	25	45	600	3.60
New York	16	49	240	6.13
Atlanta	15	22	210	2.93
Polska	12	18	132	3.00

Table 3.5: The studied network topologies.

value can be used as a weight metric to provide solutions that allow the measurement of the flows with the largest accumulated differences on their upper and lower bounds, and accordingly allows to minimize the estimation error.

Finally, please note that the here explained heuristic can be augmented with individual cost values for all resources. This allows a similar preference or discrimination of resources, for instance, due to reasons that

- specific links are more expensive to backup,
- specific nodes are more expensive to upgrade to SDN, or
- the upgrade to SDN of a node is in general more expensive than the backup of a link.

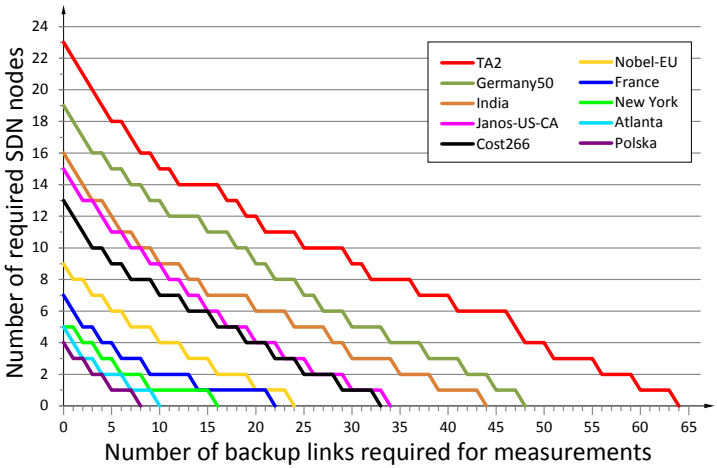


Figure 3.13: Total number of required backup links vs. SDN nodes.

The number of flow determinations of a resource has then to be divided by its cost, in order to let the heuristic choose in each iteration the resource with the biggest "return of investment".

Performance Evaluation

In our performance evaluation, we used ten topologies from the SNDlib library [36], listed in Table 3.5. We generated uniform distributed random values for the traffic matrices of each topology – which has however no impact on any of the results, as we focus only on the *number* of measurable and obtainable IE flows.

Figure 3.13 shows our main result for the ten tested topologies: the number of required SDN nodes in the network depending on the number of deployed backup links, assuming that all the resources have been located optimal. We have used the ILP model from Subsection 3.4.2 and preset the number of backup links in order to allow

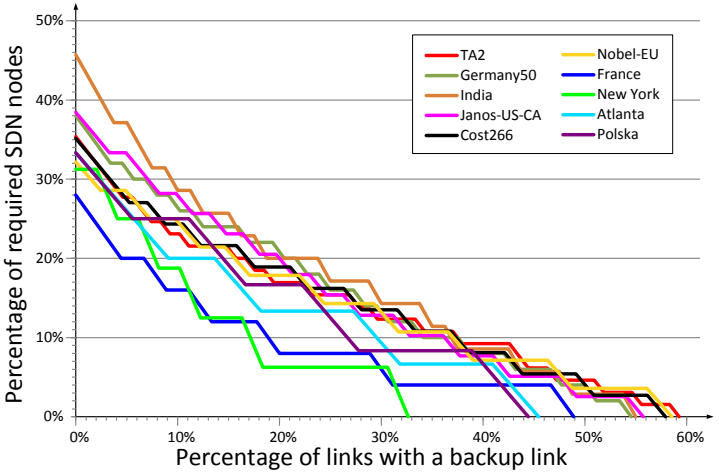


Figure 3.14: Relative interplay of backup links and SDN node quantities.

the computation of the exact number of SDN nodes for any given number of backup links. It can be seen in the figure that SDN nodes are typically traversed by a larger number of flows, which results in a relatively large number of backup links if zero SDN nodes are to be used. It can also be seen that the number of required measurement resources scales with the size of the topology.

In contrast to the absolute numbers in Figure 3.13, we rescaled the plots for Figure 3.14 to provide insights independent of the network size. The figure shows that the majority of networks exhibit a similar characteristic of requiring a relatively linear combination out of 30%-40% SDN nodes and 50%-60% backup links. There are three plots that differ slightly from this pattern, which are those of the India, France and New York topologies, which are also the three networks with the largest nodal degree (see Table 3.5) in our comparison.

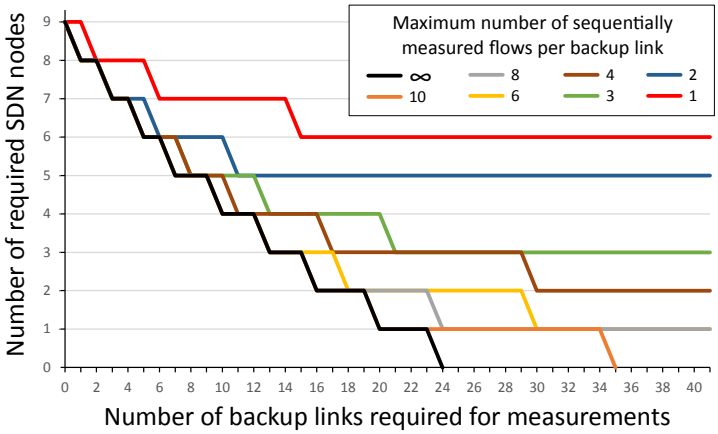


Figure 3.15: Limiting the number of sequentially measured flows per bypass.

However, the analysis of a much larger number of topologies would be required to confirm such a principle behind the observed manner. What can however be confirmed from Figure 3.14, is that the relation of required SDN nodes and backup links appear to be independent of the network size, as for instance the TA2 topology and the Nobel-EU topology (which has less than half the size of TA2) exhibit a very similar characteristic.

As discussed in Subsection 4.6.1, sequential measurements for the same traffic matrix are subject to an increased error due to non-statical traffic fluctuations. Our ILP model in Subsection 3.4.2 does therefore provide the *MaxFlows* parameter, that allows to limit the number of flows that are measured sequentially on a bypass. The shown results, however, do not consider such a limitation so far, but a large number of flows that have to be measured on the same bypass may result in a very long measurement period. Figure 3.15 shows the effect of the *MaxFlows* parameter on the number of required

SDN nodes in the Nobel-EU network. In other words, Figure 3.15 depicts how the yellow plot of Figure 3.13 changes, when the number of sequential measurements per bypass is reduced. The figure shows that a limitation to eight or less sequential measurements per bypass prevents a "full-SNMP" solution (i.e. those plots don't intersect with the x-axis), as the 756 flows of the Nobel-EU network are not distributed equally among all links. However, it can also be seen that the difference of the plots is marginal when there are four or more SDN nodes deployed (except for the limitation to a single measurement per bypass).

The heuristic algorithm can be used instead of the linear optimization model in case finding the optimal solution exceeds acceptable computation times due to the network's size. We propose for accuracy to use the heuristic only for a subset of the required resources (i.e., to terminate the heuristic before the solution is complete), and to preconfigure the ILP model with the chosen resources to find the remaining resources.

Figures 3.16 and 3.17 show the total number of required resources (solid line, left y-axis), depending on to what extent the problem was solved with the heuristics (x-axis), before the remaining resources were determined with the ILP model. The less resources are heuristically determined, the larger is the time complexity of the remaining linear optimization problem, which can be observed in the second plot (dashed line, right y-axix) in both figures: While the heuristic (compared to the ILP) terminates in negligible time[3], the time to find the optimal locations of the remaining resources increases beyond exponentially with the number of those resources.

We show these two results for different purposes: Figure 3.16 shows the discussed behavior for the TA2 topology, which is the largest out of the ten compared ones, and thus the most demanding

[3]The required computation time of our heuristic was below five seconds in a 1000 node random topology that we additionally tested.

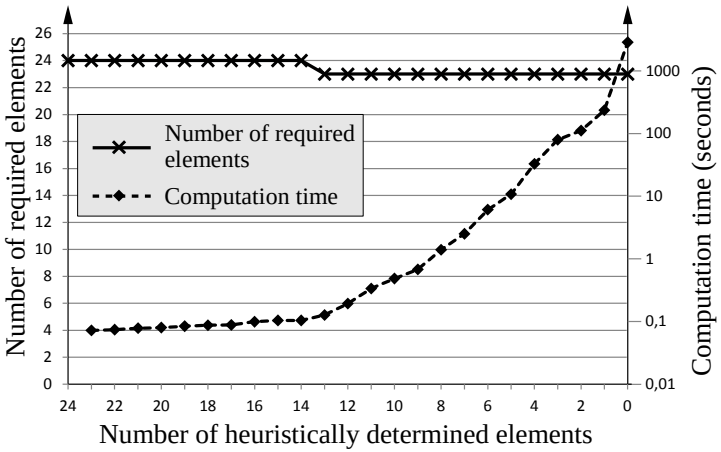


Figure 3.16: Performance of the greedy algorithm in the TA2 topology.

in terms of the optimality. Especially the last four data points of the time plot show that the time complexity becomes prohibitive large. It should be noted that the initialization of the optimization model requires a fixed duration depending on the network size and independent of the actual problem size, which is why the time values in the x-axis range between 23 and 14 preset resources appear to be somewhat constant. The 65 nodes of the TA2 topology can therefore be considered as borderline tractable regarding time complexity for the ILP.

Figure 3.17 shows the same result for the Janos-US-CA topology, which we chose because we observed the comparably worst performance of the heuristic amongst the tested topologies: The heuristic alone determines 18 resources for the complete traffic matrix (the leftmost data point), whereas the optimal solution requires only 15 resources (the rightmost data point). This suggests that the heuristic

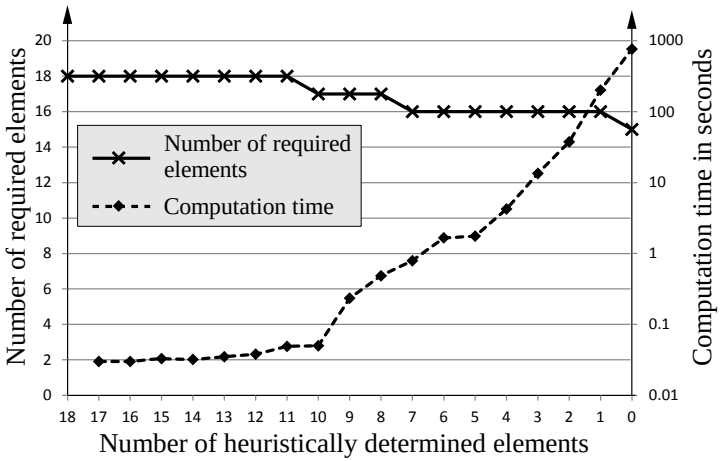


Figure 3.17: Performance of the greedy algorithm in the Janos-US-CA topology.

should only be used to the point where the search for the remaining resources by an ILP solver is acceptable, for instance, by iteratively reducing the number of heuristically pre-configured resources.

3.4.3 Compatibility of the two Node Placement Strategies

This subsection compares the two SDN node deployment strategies from the previous Subsections 3.4.1 and 3.4.2 and analyzes the compatibility of their objectives. While the deployment of backup links by a network operator solely for the purpose of traffic measurements still appears somehow comprehensible, we assume that the deployment of SDN nodes for the same purpose (like proposed in Subsection 3.4.2) is rather unrealistic due to the required cost and infrastructure upgrade effort. We have therefore tested to what extent traffic measurements can benefit from a more realistic upgrade

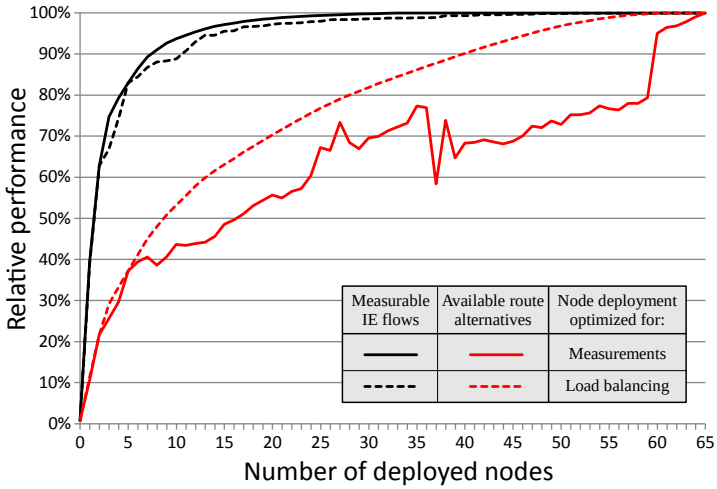


Figure 3.18: Compatibility of the two node deployment strategies.

strategy, like the one in Subsection 3.4.1. The objective of that strategy can be described as providing the maximum control on routing decisions to the central SDN controller for a given number of SDN nodes, which is here measured in *number of route alternatives*. It was shown in the performance evaluation of Subsection 3.4.1 that a larger total number of available paths to chose from allows for a more sophisticated traffic engineering and load balancing of the network, which appears to be the most desirable objective for network operators.

Figure 3.18 plots the two performance measures, i.e., number of alternative routes (the red lines) and number of measurable flows (the black lines) depending on the number of SDN nodes, using either the realistic upgrade to SDN strategy from Subsection 3.4.1 (the dashed lines) or the locations optimal for measurements (the solid lines). We here used the TA2 topology, which due to its size

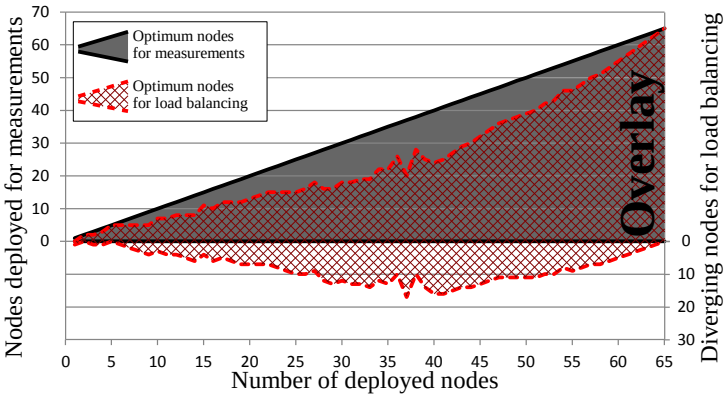


Figure 3.19: Overlap of chosen nodes of both deployment strategies.

(65 nodes) provided the largest resolution of the x-axis, and we deployed solely SDN nodes (and no backup links) in order to make the plots comparable. We furthermore normalized all values with the respective maxima (i.e., 16856 alternative routes after full SDN deployment vs. a total of 4160 IE flows) and show only the relative performance on the y-axis. A comparison of the two black plots shows that the SDN upgrade strategy that maximizes routing control provides near optimal locations for measurements, as the number of measurable IE flows falls negligibly below the ones that are achievable with optimally located SDN nodes. It can furthermore be seen that the reverse (i.e. comparing the two red plots) does not hold: the node locations optimal for traffic measurements are significantly less suited for traffic engineering and load balancing. An important finding of our work is thus that operators considering to upgrade their legacy IP networks to SDN can use the strategy from Subsection 3.4.1 without noticeable drawbacks on SDN's traffic measurement capabilities. The chosen node locations can then be preset in the here presented ILP and heuristic to determine solely

the missing backup links to complete the traffic matrix.

The two strategies choose indeed very similar nodes, which we attempt to visualize in Figure 3.19. The figure shows the overlap of nodes chosen from both strategies for a given number of deployable SDN nodes. The gray area depicts the optimal nodes for traffic measurements, whereas the red shaded area shows the optimum nodes for load balancing and traffic engineering. The two bounding (dashed red) lines of that area can be interpreted as following: The upper line plots how many of the nodes optimally deployed for load balancing are also optimal for measurements (left y-axis). The lower line plots the number of nodes optimally deployed for load balancing that have not been chosen by our measurement location optimization (right y-axis).

3.5 Capacity Dimensioning

Capacity dimensioning is an important network management task, where the links of a network are re-dimensioned by the operator to accommodate the network capacity to the changing traffic demands for the next planning period. Shortest path routing would always result in minimum capacity requirements, if link capacities could be commissioned in arbitrary sizes, but link capacities are available in our assumed transport network architecture only in fixed granularities (i.e., 10 Gbit/s, 40 Gbit/s, and 100 Gbit/s) and with each capacity type there is a cost associated for the required devices in the optical layer (i.e., transponders, line cards, energy, etc.). However, our model does not comprehend other constraints of actual switching hardware, like the possibility of port aggregation, or the fact that line cards have a defined configuration of ports (i.e. ports are not available per size in arbitrary numbers). Capacity planning is trivial in OSPF networks, when the routing remains unchanged and the traffic demand is known for the next planning period: if the

estimated traffic load on a link exceeds a certain utilization threshold, the link is provisioned with the next larger capacity granularity. However, more sophisticated control planes allow to steer the routing to avoid underutilized links (and thus capacity wasting), which for instance occur in case of an estimated demand of 11 Gbit/s in the next planning period on a link that was previously provisioned with 10 Gbit/s capacity: This link requires an upgrade to 40 Gbit/s and will exhibit a poor utilization of less than 28% during the next planning period. A sophisticated control plane thus routes the traffic flows such that poorly utilized links are avoided, which allows to reduce the total capacity requirements to the actual demand.

3.5.1 ILP Model

I will now explain the mathematical model for capacity planning in full SDN, SDNp, and non-SDNp hybrid SDN/OSPF networks. The objective of this model is to determine the traffic routing that requires the minimum total cost for the provisioned link capacities, while it suffices all architectural constraints. The notation of the model is shown in Table 3.6.

The objective function of this model minimizes the summation of the cost for the commissioned capacity types of all links in the network:

$$\text{Minimize} \qquad \sum_{\ell \in \mathcal{L}'} \sum_{t \in \mathcal{T}} \psi(\ell, t) \cdot cost(t) \tag{3.19}$$

subject to all constraints explained below. The first constraint limits the amount of traffic on any link $\ell \in \mathcal{L}'$ to the commissioned link capacity in consideration of the maximum allowed link utilization $u_{\max}$:

$$\forall \ell \in \mathcal{L}' : \sum_{p \in \mathcal{P}} \rho(p) \cdot \mathcal{R}_\ell^p \cdot dem(p) \leq \sum_{t \in \mathcal{T}} \psi(\ell, t) \cdot cap(t) \cdot u_{\max} \tag{3.20}$$

We need a routing constraint to assure that exactly one path will

Parameters

Set	Meaning
$\mathcal{N}$	All nodes in the network
$\mathcal{L}'$	All directional links in the network
$\mathcal{F}$	All traffic flows in the network
$X_k \subseteq X$	The subset of X with all elements of the k^{th} sub-domain
$\overline{\mathcal{N}}_k$	All nodes in $\mathcal{N} \backslash \mathcal{N}_k$
$\mathcal{T}$	All link capacity types
$\mathcal{P}$	All routing paths in the network
M_k	All metric vectors $\vec{m}$ that can be advertised by the border nodes of the k^{th} sub-domain

Real	Meaning
$cost(t)$	The cost associated with a link of capacity type $t \in \mathcal{T}$
$dem(f)$	The demand (traffic load) of flow $f \in \mathcal{F}$
$u_{\max}$	The maximum allowed link utilization ($0 \leq u_{\max} \leq 1$)

Integer	Meaning
K	The number of sub-domains of the network
$cap(t)$	The capacity of type $t \in \mathcal{T}$

Boolean	Meaning
$\mathcal{R}_\ell^p$	Path $p \in \mathcal{P}$ traverses link $\ell \in \mathcal{L}'$
$cor(f, p)$	Flow $f \in \mathcal{F}$ corresponds to path $p \in \mathcal{P}$
$cons(p, \vec{m})$	Usage of path p is consistent with the advertisement of $\vec{m}$
$dst(p, d)$	Node $d \in \mathcal{N}$ is the destination of path $p \in \mathcal{P}$

Variables

Boolean	Meaning
$\psi(\ell, t)$	Capacity type $t \in \mathcal{T}$ is used on link $\ell \in \mathcal{L}'$
$\varphi(\vec{m}, d)$	Metric vector $\vec{m} \in M$ is advertised for destination $d \in \mathcal{N}$
$\rho(p)$	Path $p \in \mathcal{P}$ is used

Table 3.6: Notation for capacity dimensioning.

be used for each flow f (which corresponds to a specific source-destination pair of nodes) in the network, i.e.,

$$\forall f \in \mathcal{F} : \sum_{p \in \mathcal{P}} \rho(p) \cdot cor(f,p) = 1 \tag{3.21}$$

Please note that all possible routing paths are precomputed according to the individual scheme's inherent working principle and subsumed in $\mathcal{P}$, in order to reduce the complexity of this ILP model.

The following two constraints are specifically for SDNp and deal with issues of conformity of the used routing paths with LSAs advertised by the SDN controller. Therefore, the solution space of full SDN and regular (non-SDNp) hybrid SDN/OSPF is not constraint by the remaining two inequations. The first SDNp routing constraint guarantees that the choice of paths to be used is segment-wise (i.e., per sub-domain) consistent with the link-weight-based routing behavior of OSPF routers:

$$\begin{gathered} \forall k \in [1,K],\ \forall p \in \mathcal{P},\ \forall d \in \overline{\mathcal{N}}_k : \\ dst(p,d) \cdot \rho(p) \leq \sum_{\vec{m} \in M_k} \varphi(\vec{m},d) \cdot cons(p,\vec{m}) \end{gathered} \tag{3.22}$$

The precomputed parameter $cons(p,\vec{m})$ is 1 only if the advertisement of link metrics in $\vec{m}$ in sub-domain k for destination d leads to an OSPF forwarding behavior such that the exit border node in k is contained in path p, and 0 otherwise. Finally, we need a constraint to assure that in each SDNp sub-domain exactly one metric vector is advertised for each sub-domain-external destination:

$$\forall k \in [1,K],\ \forall d \in \overline{\mathcal{N}}_k : \sum_{\vec{m} \in M_k} \varphi(\vec{m},d) = 1 \tag{3.23}$$

3.5.2 Performance Evaluation

For the performance analysis, we used the Cost266, the Janos-US-CA, and the Nobel-EU topologies from the SNDlib library [36]. Figure 3.20 shows the results of capacity planning in relation to the requirements of an OSPF-controlled network. OSPF is taken as worst

case, as no load balancing is applied, whereas all other operational schemes allow to optimize the routing to improve resource utilization, which in turn allows to reduce the link capacities. Figure 3.20 compares the capacity requirements in the three different network topologies depending on the used control scheme, whereas the SDNp results are furthermore classified depending on the actually applied partitioning into sub-domains of the initial topology. The Cost266 topology partitioned into 2, 4, and 10 sub-domains is depicted in Figure 3.5, the Janos-US-CA topology partitioned into 2, 4, 6, and 10 sub-domains is depicted in Figure 3.4, and the Nobel-EU topology partitioned into 2, 4, and 6 sub-domains is depicted in Figure 3.3. We compare the performance of SDNp furthermore with full SDN deployment and the regular (non-SDNp) hybrid SDN/OSPF control plane scheme, where all nodes participate in OSPF, and hybrid nodes can additionally be configured dynamically with high priority routing rules. For this scheme we assumed that (at least) 50% of all nodes are SDN-enabled and the optimal locations of these nodes were determined based on the location optimization method in Subsection 3.4.1. The actual number of SDN-enabled and legacy OSPF nodes is given in the second and third column of Figure 3.20.

The evaluation of capacity requirements was carried out as follows: We used the initially unpartitioned network for the "OSPF" case, assigned uniform link metrics, and determined the OSPF least cost paths, which resulted in minimum hop count routing. We then assigned uniformly distributed traffic demands to all source-destination pairs in the network and rescaled them all with the same scaling factor, such that the maximum link load is set to 80 Gbit/s. We assigned minimum capacities to all links such that no link exceeds 80% utilization. We assumed that link capacities are available in granularities of 10 Gbit/s, 40 Gbit/s, and 100 Gbit/s. All other results in Figure 3.20 show the minimum capacity requirements of the according control plane scheme after the routing has been opti-

Control Plane Scheme	SDN nodes	OSPF nodes	Required Capacity
Cost266: 37 nodes, 57 links, 1332 SD flows			
OSPF	0	37	100%
SDN Partitioning (2 Sub-Domains)	4	33	77,1%
SDN Partitioning (4 Sub-Domains)	7	30	72,6%
Stacked Hybrid, 50% SDN	19	18	70,5%
SDN Partitioning (10 Sub-Domains)	11	28	65,4%
Complete SDN Deployment	37	0	63,3%
Janos-US-CA: 39 nodes, 61 links, 1482 SD flows			
OSPF	0	39	100%
SDN Partitioning (2 Sub-Domains)	4	35	72,5%
Stacked Hybrid, 50% SDN	20	19	70,5%
SDN Partitioning (4 Sub-Domains)	8	31	68,7%
SDN Partitioning (6 Sub-Domains)	10	29	66,9%
SDN Partitioning (10 Sub-Domains)	12	27	65,8%
Complete SDN Deployment	39	0	62,0%
Nobel-EU: 28 nodes, 41 links, 756 SD flows			
OSPF	0	28	100%
SDN Partitioning (2 Sub-Domains)	3	25	78,4%
Stacked Hybrid, 50% SDN	14	14	71,8%
SDN Partitioning (4 Sub-Domains)	5	23	69,6%
SDN Partitioning (6 Sub-Domains)	7	21	67,1%
Complete SDN Deployment	28	0	65,6%

Figure 3.20: Required link capacities in the different topologies.

mized under the schemes' individual routing constraints. The first noticeable characteristic of this result is that *all* evaluated schemes are able to save considerable amounts of link capacity compared to OSPF. This was however to be expected, as link utilization is not considered in the routing algorithm of OSPF, which thus can lead to significant capacity wasting. Packets are solely routed via shortest (i.e., least metric cost) paths, which may result in a link load that just slightly exceeds a particular capacity granularity. More remarkable is however, to what extent SDNp can keep up with or even

outperform the 50% and 100% SDN deployment. It can be seen in the figure that SDNp with the most sub-domains is relatively close to the result of full SDN deployment in each tested topology. We generally conclude that a migration to SDN-enabled devices beyond the requirements of SDNp (with small sub-domains) can not result in significant further capacity savings, whereas a homogeneous SDN deployment doubtlessly provides other operational and management advantages. Another remarkable outcome of this evaluation is the fact that SDNp outperforms stacked hybrid SDN/OSPF operation with only a fraction of the required SDN-enabled nodes. It can finally be seen that even the partitioning into only two sub-domains can considerably improve resource utilization compared to plain OSPF, while the number of required SDN nodes is notably low. Please note that the results of SDNp shown in Figure 3.20 are suboptimal in the sense that a joint optimization of partitioning *and* link dimensioning could further improve the results. Such an approach was however neglected due to its computational complexity.

3.6 Fault Tolerance

Fault tolerance of a network refers to the capability to continue operation under failure conditions. This does not only require the ability to quickly shift operations away from the failed network resources, but also to have sufficient reserve capacities available at the network resources that take over these operations. This section addressees the network planning task of determining the network capacity to the changing traffic demands and events like sudden traffic surges and possible network failures for the next planning period. Like depicted in Figure 3.21, fault tolerant capacity dimensioning is a planning process that requires a demand forecast, which extrapolates the demand at the end of the next planning period based on the current demand (which is monitored constantly by the opera-

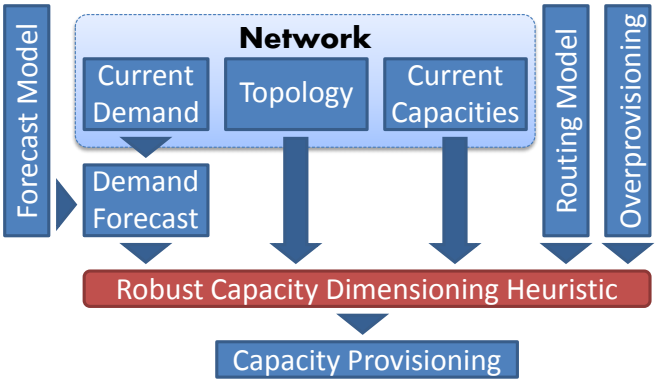


Figure 3.21: Fault tolerant capacity planning.

tor) and a forecast model (i.e., statistical methods to estimate the increase of traffic demands based on historical monitoring data). Fault tolerant capacity planning additionally requires topology and capacity information from the network, which is typically available from the network management system in place. Finally, mapping the forecast demand to the links based on the routing model is an optimization process with the objective to minimize the cost of the required capacity upgrades.

The traffic demand in the here targeted Internet backbone networks is subject to two kinds of variations relevant for demand forecasting, which are defined by the observed time frame: a) Daily pattern: Traffic variations on backbone links show a strong daily pattern with typically low utilization in the early morning and maximum utilization during the evening. b) Annual increase: Traffic demands increase in the long term on average with an annual growth rate that is known to be relatively predictable at least for the next few years. Traffic forecasting has to take both time scales into consideration: The future size of a traffic flow is estimated based on its

current daily maximum (a), and upscaled with the annual increase rate (b) till the end of the next planning period. A more sophisticate approach may take flow characteristics of individual demands into consideration, which is however here out of scope, and we assume that we have an exact demand forecast available as input data for the capacity dimensioning process.

Demand measurements are always averaged over a specific sample interval (e.g., five minute averages). They therefore lack information on the variation *within* each interval that is caused by micro bursts. These bursts can cause short-term congestions, which in turn cause jitter, increased delay, or even packet loss, even though the link may not be highly utilized on average. The relation between the average link load and the required link capacity – that reduces the frequency of short-term congestions according to the targeted level of network service quality – is referred to as the overprovisioning factor.

The routing model to be used in capacity planning needs to reflect the actual routing configuration capabilities of the network. In case of fixed (i.e., non-dynamic) OSPF link metrics (which is the common case and assumed in our OSPF model), routing changes in an OSPF network are completely predetermined for all network failures, whereas dynamic reactions on sudden traffic changes are impossible. Our OSPF model therefore represents the absolute zero on network programmability. A complete SDN deployment, on the other hand, provides complete freedom regarding the configuration of routing paths, which allows to efficiently load-balance the traffic. SDN therefore represents the full level of network programmability (along with other networking schemes like MPLS or Policy Based Routing). Hybrid SDN/OSPF operation provides routing configuration capabilities somewhere between these two levels, depending on the number of SDN nodes and their locations, and the used routing model (i.e., regular hybrid or SDNp). A valid routing path in such a network is a concatenation of OSPF paths and SDN links,

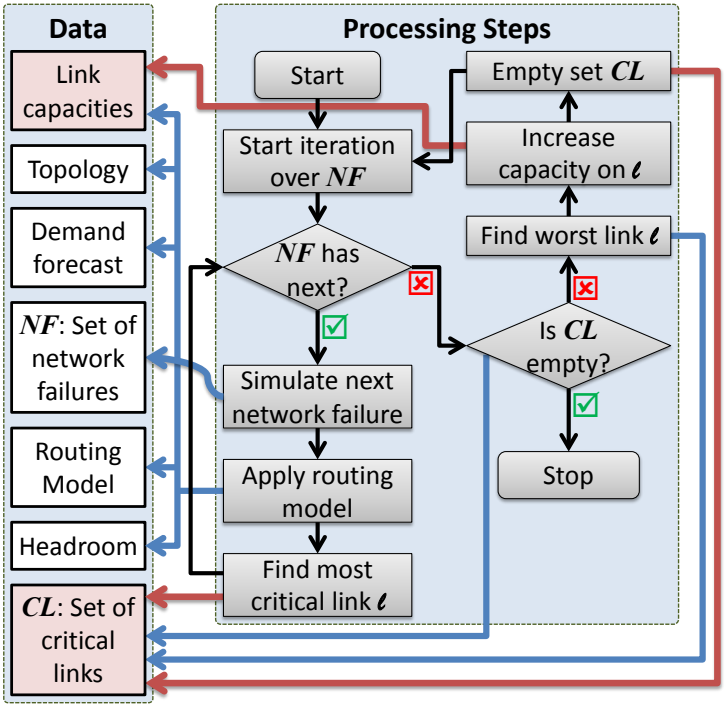


Figure 3.22: The heuristic algorithm for robust link capacity dimensioning.

like detailed in Sections 2.4 and 2.5 respectively.

3.6.1 Algorithm

We use a simple heuristic based on an iterative greedy algorithm to determine sufficient link capacities in respect of a predefined set of network failures and the capabilities of the deployed control plane to reroute traffic. The algorithm is depicted as a flow chart in Fig-

ure 3.22, where the boxes in the *Processing Steps* container represent algorithmic states and the boxes in the *Data* container represent working copies of information retrieved from the network management system like explained above. We used black arrows to indicate state transitions of the algorithm, red arrows to indicate write access of a processing step on a data set, and blue arrows to indicate read access. The algorithm consists of two stacked iterations: The inner one iterates over all network failures (depicted as the set NF) and applies the routing model to each resulting network scenario, i.e., it tries to load balance the forecast load (based on the traffic engineering model explained in Section 4.3) in the network, assuming that the particular failure occurred, and determines the most critical link (which is the one with the largest overload). That link is stored for each failure scenario in the set CL. After the inner loop has iterated over all network failures, the algorithm performs one cycle of the outer iteration, where the worst case of all critical links in CL is chosen to be capacity increased (for now only in the working copy of the algorithm) to the next capacity granularity (e.g., from 10 Gbps to 40 Gbps). The algorithm automatically stops when the working copy of the link capacities have been increased to the point where the routing scheme can handle all network failures without overload on any link. The output of the algorithm is the working copy of link capacities that now contains the desired link capacities required for redimensioning the network.

3.6.2 Performance Analysis

For our performance analysis, we used the Nobel-EU (28 nodes, 41 links), the Cost266 (37 nodes, 57 links), and the Janos-US-CA (39 nodes, 61 links) topologies from the SNDlib library [36]. Figure 3.23 shows the results of our first experiment, where we used our heuristic for robust link capacity planning. All results are normalized with the capacity requirements of an OSPF-controlled network. Native OSPF

is taken as reference scenario, as its routing model provides no load balancing whatsoever, whereas all other operational schemes allow to optimize the routing to improve resource utilization and to reroute traffic more efficiently in case of a failure, which in turn allows to reduce the required link capacities. We compare the capacity requirements of OSPF, full SDN deployment, and the two hybrid control planes: stacked hybrid and SDN Partitioning, whereas the latter is furthermore classified depending on the number of sub-domains in which the initial topology was partitioned. The Cost266 topology was partitioned into 2, 4, and 10 sub-domains, the Janos-US-CA topology was partitioned into 2, 4, 6, and 10 sub-domains, and the Nobel-EU topology was partitioned into 2, 4, and 6 sub-domains.

For the regular (non-SDNp) hybrid scheme we assumed that (at least) 50% of all nodes are SDN-enabled and the optimal locations of these nodes were determined based on the location optimization method in Subsection 3.4.1. The actual number of SDN-enabled and legacy OSPF nodes is given in Figure 3.23. The overlaid (darker) bars show the minimum capacity requirements of a routing scheme without any provisions for network failures. These values have been taken to initialize the capacity planning heuristic (Fiure 3.22). The capacity requirements for fault tolerant operation determined by the heuristic under consideration of all single fiber cuts in the network[4] are shown as the bars in brighter colors, including the relative requirements (in percent) compared to OSPF. Traffic was assumed to be uniformly distributed among all node pairs scaled such that the maximum link load in the OSPF case without link failures was 40 Gbps. Link capacities were available in 10 Gbps, 40 Gbps, and 100 Gbps.

It can be seen from Figure 3.23 that all hybrid schemes require significantly less capacities than native OSPF for fault tolerant op-

[4]There are many other possible failure scenarios than a single fiber cut, e.g. node failures, cyber attacks, etc., which have however not been further considered in the results.

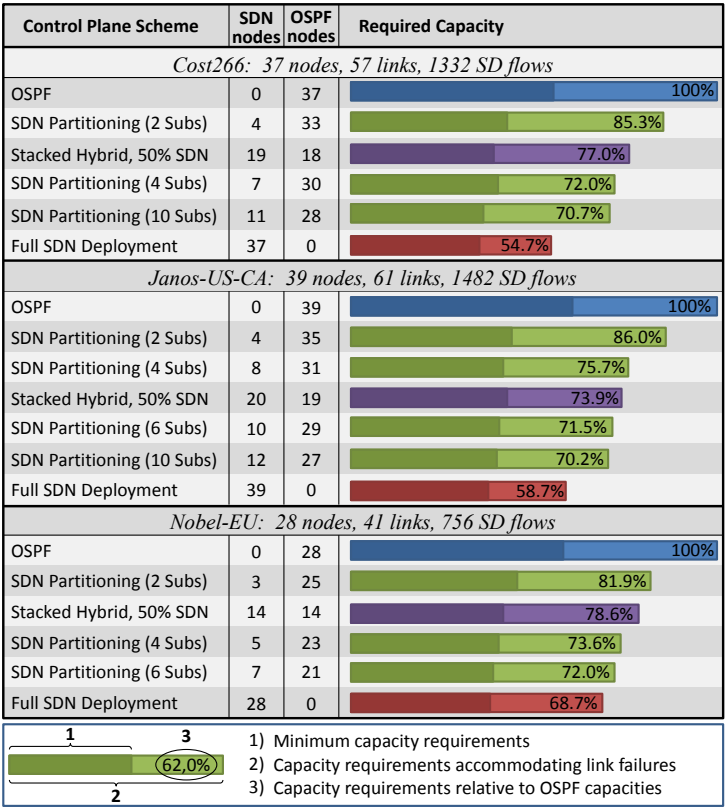


Figure 3.23: The capacity requirements to accommodate link failures.

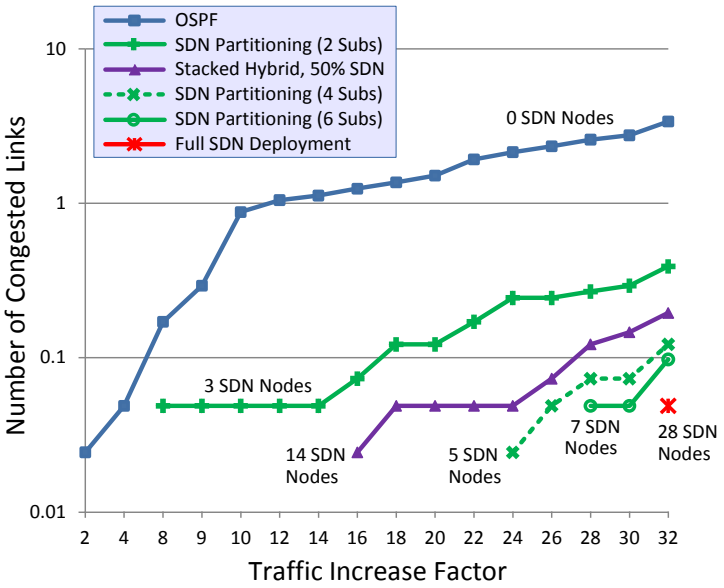


Figure 3.24: Number of congested links in case of sudden traffic surges.

eration, whereas SDN Partitioning requires significantly less SDN-enabled devices to be deployed in the network to achieve results comparable to the stacked hybrid scheme. Even very few SDN nodes operated in SDN Partitioning mode suffice to provide a level of routing control that clearly reduces the capacity requirements. Finally, our results suggest that our method is unsusceptible against the topology of the network, considering the similarities of the results in the three different networks.

Figure 3.24 shows our second experiment, in which we analyzed the behavior of the different routing schemes when sudden traffic surges occur in the coincidental case of a single fiber cut in the net-

work. We here used the Nobel-EU topology with link capacities dimensioned for fault tolerant OSPF operation, and increased the traffic between the two node pairs Madrid - Stockholm and Athens - Glasgow in both directions. The node pairs have been chosen such that they are most distant (geographically and in terms of hop count), thus we stressed the network with four sudden elephant flows with each of them traversing the complete diameter of the network. The original traffic flows between these node pairs were increased with the scaling factors given at the x-axis of Figure 3.24, and the y-axis (in base-10 log scale) shows the probable number of congested links. The result of this experiment confirms what the previous experiment suggested: The level of routing control in hybrid SDN/OSPF networks provides a significant advantage over native OSPF operation without the investments required for a full SDN deployment. Again, SDN Partitioning outperforms the other hybrid mode with comparably few SDN nodes. It can be seen that the capability of a network operated in native OSPF can not handle elephant flows properly and the probability of congested links in case of a link failure is increasing rapidly with the size of the flows, which suggests that traffic forecasts should rather be upscaled significantly before capacity planning is carried out. Full SDN deployment, on the contrary, appears to be unsusceptible to traffic surges, as in our experiment it required to scale up the four original flows with factor 32 to see at least any congestion.

3.7 Summary

The new hybrid SDN/OSPF control plane SDNp, which is proposed in this thesis, can establish a centralized control over the distributed routing protocol by partitioning the topology into sub-domains with SDN-enabled border nodes. The key characteristic of this approach is that update messages of the legacy routing protocol have to tra-

verse SDN border nodes to reach neighboring sub-domains. This allows the central controller to modify how sub-domains view one another, which in turn allows to steer inter-sub-domain traffic. The approach allows to trade off the degree of dynamic control against the simplicity of OSPF by means of network clustering. The mathematical model of network clustering, that was explained in this chapter, allows to adjust the size of the sub-domains exactly to the operator's requirements with a minimum requirement regarding the number of SDN-enabled nodes.

We have also shown in this chapter to what extent the placement strategy for SDN-enabled routers and backup links in hybrid SDN/OSPF can solve the IP traffic matrix and related monitoring problem, which is inherent to the IP layer. We therefore provided a linear optimization model and a heuristic algorithm for optimum SDN node and backup link placement, which assures the retrieval of the full traffic matrix under minimum resource requirements. In this novel approach, the IP traffic matrix is generated from measurements of individual ingress-egress flows using both types of byte counters, from backup links between legacy routers and flow table entries of OpenFlow-enabled routers. Instead of using expensive monitoring infrastructure for non-SDN devices, we proposed to use policy based routing for backup ports and SNMP-based byte counters, features that are likely to be readily available in IP networks. The numerical evaluation showed that there is a near linear trade-off between SDN nodes and backup links that are required for a full traffic matrix, which lets us conclude that a hybrid network with a few SDN nodes can already provide complete traffic statistics, when enough backup links are available for SNMP-based measurements. We have shown in our analysis that the proposed SDN deployment strategy for traffic measurements in hybrid networks is very compatible with SDN upgrade strategies that aim for maximum network control.

This chapter finally detailed how to model both hybrid SDN/OSPF network architectures, non-partitioned and SDNp, for the optimization of the most important network planning operations. Based on these models, we numerically evaluated the performance of SDNp in comparison to OSPF operation, full SDN deployment, and regular hybrid SDN/OSPF (assuming a 50% SDN deployment) without partitioning. The analysis of the required network capacities and the minimum provisions for fault tolerant operation showed that – depending on the degree of partitioning – the resource requirements of SDNp range between regular (non-partitioned) hybrid SDN/OSPF with 50% SDN nodes and full SDN deployment, but with relatively few SDN-enabled routers. The low number of required SDN routers is the feature od SDNp, which allows that a relatively high number of nodes can remain in the *configure-once-never-touch-again* operation, which is a known and desired feature of OSPF. Finally, SDNp showed superior performance in almost all cases of the carried out experiments, outperformed only by full SDN deployment, which not only eliminates legacy protocols, which no operator can easily commit to, but also requires significant investments in new networking equipment.

We conclude that SDNp provides a pragmatic and efficient migration path for network operators, as an initial partitioning into two sub-domains requires only a few SDN nodes. Sub-domains could iteratively be partitioned into smaller sub-domains in further migration steps, which would gradually increase the central control for routing with moderate investments in new networking equipment.

4

Operation of Hybrid Legacy/SDN Networks

4.1 Introduction

Operations, administration and maintenance (OAM) is a networking term that refers to the key functionalities for a reliable operation of a network, involving standards, tools, processes, and activities for the configuration, monitoring, and fault recovery of networks. This chapter addresses the modeling of OAM for the IP layer of Internet backbone networks in general, and accounts in particular for hybrid SDN/OSPF control plane models, which are new. Many aspects of OAM have been automated in the IP layer as an integral part of the distributed protocols like OSPF, which not only autonomously carries out continuity checks (denoted as OSPF Hello packets), but more importantly reacts on network failures with an automatic recovery process. IP layer protocols provide however only basic OAM functions for network operators, like SNMP or ICMP (which is used by the simple diagnostic tools Ping and Traceroute), and any advanced OAM mechanism (e.g. Cisco's PBR or Juniper's Jflow) is, if implemented, accessible only via vendor dependent APIs. SDN, on the other hand, is vendor-neutral and based on open standards, like the prevalent OpenFlow protocol for the communication between the central controller and the elements of the forwarding plane. Moreover, the most popular SDN controller implementations provide an

open northbound API that allow for own OAM function implementations customized for the individual requirements of network operators.

The simultaneous operation of a distributed routing protocol and a central SDN controller is referred to as a hybrid control plane. This chapter analyzes to what extent a hybrid SDN/OSPF network can provide OAM functions and how such a network can be modeled to allow for the optimization of network operational tasks like traffic engineering and failure recovery. An important focus of this chapter is, like in the previous chapter, SDN Partitioning, which is a novel mode of operation for hybrid SDN/OSPF networks, proposed in this thesis. This chapter furthermore addresses the question of how to efficiently schedule network operations, and how to exploit the new traffic monitoring capabilities provided by OpenFlow-conform devices, in order to solve the traffic matrix problem, which is inherent to the IP layer.

4.2 Supporting Publications

1. M. Caria and A. Jukan, "On the IP traffic matrix problem in hybrid SDN/OSPF networks," *submitted for review to IEEE Transactions on Network and Service Management.* (Preprint available at arXiv.org: `https://arxiv.org/abs/1610.08256`)

2. M. Caria, A. Jukan, and M. Hoffmann, "SDN Partitioning: A centralized control plane for distributed routing protocols," *IEEE Transactions on Network and Service Management,* Volume 13, Number 3, pp. 381-393, September 2016. (Preprint available at arXiv.org: `https://arxiv.org/abs/1604.04634`)

3. M. Caria and A. Jukan, "The perfect match: Optical bypass and SDN Partitioning," in *2015 IEEE 16th International Conference on High Performance Switching and Routing (HPSR),*

July 2015, pp. 1-6.

4. M. Caria, T. Das, and A. Jukan, "Divide and Conquer: Partitioning OSPF networks with SDN," in *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM),* May 2015, pp. 467-474. (Preprint available at arXiv.org: `https://arxiv.org/abs/1410.5626`)

5. M. Caria and A. Jukan, "A novel approach to accurately compute an IP traffic matrix using optical bypass," in *2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013),* May 2013, pp. 1135-1141.

6. M. Caria, A. Engelmann, A. Jukan, and B. Konrad, "How to slice the day: Optimal time quantization for energy saving in the Internet backbone networks," in *Global Communications Conference (GLOBECOM), 2012 IEEE,* December 2012, pp. 3122-3127.

4.3 Traffic Engineering

Traffic engineering is an important network management task, where the routing of traffic flows is changed for two different purposes: to meet customers' QoS requirements, and to reroute excess traffic away from over-utilized network resources to resources that have sufficient provisions to accommodate the said traffic. The second purpose is commonly referred to as load balancing and in the focus of this section.

4.3.1 Mathematical Model

The mathematical model for traffic engineering in SDN-partitioned OSPF networks, full SDN networks, and regular (non-SDNp) hybrid SDN/OSPF networks is explained here. Whereas OSPF is used in

Parameters

Set	Meaning
$\mathcal{N}$	All nodes in the network
$\mathcal{L}'$	All directional links in the network
$\mathcal{F}$	All traffic flows in the network
$X_k \subseteq X$	The subset of X with all elements of the k^{th} sub-domain
$\overline{\mathcal{N}}_k$	All nodes in $\mathcal{N} \backslash \mathcal{N}_k$
$\mathcal{P}$	All routing paths in the network
M_k	All metric vectors $\vec{m}$ that can be advertised by the border nodes of the k^{th} sub-domain
$\mathcal{Y}$	The linear functions y that jointly resemble a lower bound of a quadratically increasing utilization cost

Real	Meaning
$dem(f)$	The demand (traffic load) of flow $f \in \mathcal{F}$

Integer	Meaning
K	The number of sub-domains of the network
$cap(\ell)$	The capacity of link $\ell \in \mathcal{L}'$

Boolean	Meaning
$\mathcal{R}_\ell^p$	Path $p \in \mathcal{P}$ traverses link $\ell \in \mathcal{L}'$
$cor(f, p)$	Flow $f \in \mathcal{F}$ corresponds to path $p \in \mathcal{P}$
$cons(p, \vec{m})$	Usage of path p is consistent with the advertisement of $\vec{m}$
$dst(p, d)$	Node $d \in \mathcal{N}$ is the destination of path $p \in \mathcal{P}$

Variables

Real	Meaning
$\kappa(\ell)$	Utilization cost assigned to link $\ell \in \mathcal{L}'$

Boolean	Meaning
$\varphi(\vec{m}, d)$	Metric vector $\vec{m} \in M$ is advertised for destination $d \in \mathcal{N}$
$\rho(p)$	Path $p \in \mathcal{P}$ is used

Table 4.1: Notation for capacity dimensioning.

the numerical evaluation for comparison, it is not modeled in this section, as distributed routing protocols do not provide dynamic load balancing. We use the notation shown in Table 4.1. The formulation of a linear objective function for load balancing is not without issues and there exist various approaches in the literature: The minimization of the maximum link utilization is one of the common models. However, it was discussed in [44] that this approach yields poor results in case of unavoidable bottlenecks: When there is a heavy loaded link that can not be relieved during the optimization, the objective to minimize the maximum link utilization doesn't load balance less loaded links at all. This can be avoided with a cost (i.e., penalty) function that increases quadratically with the link utilization. Every link can then be associated with a cost according to its utilization and the objective of the optimization is then to minimize the total cost in the entire network.

As ILP models require linear constraints, a quadratically increasing cost function must be emulated with a piecewise linear function (i.e., with a concatenation of straight lines), like proposed in [44]. The set $\mathcal{Y}$ of functions $\{y_0, y_1, ...\}$ that is used in this thesis is shown in Figure 4.1, and the intention behind it is as follows. Links with a utilization below 50% are considered non-critical, and such links therefore generate zero cost. The closer the link utilization gets to 100%, the more sensitive the link gets to traffic bursts that could cause congestion, i.e. the *criticality* of a link is increasing with its utilization. The straight lines of $\mathcal{Y}$ are fixed such that each of them constitutes the lower bound of the valid cost region for a 5% section on the x-axis, and the gradient increases stepwise. The link utilization is evidently bounded by 0% and 100%, and we constrain the utilization cost for each link to be *greater equal* than all cost functions. The resulting valid solution space is depicted in Figure 4.1.

The objective function then minimizes the summation of the uti-

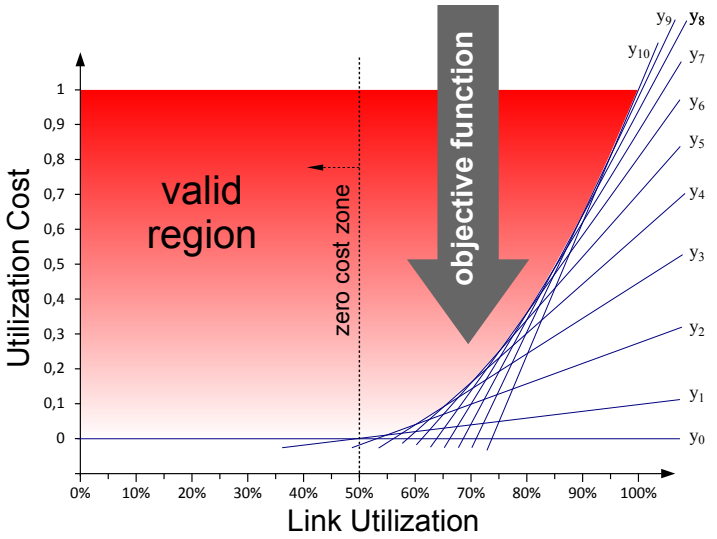


Figure 4.1: Emulation of a quadratically increasing cost function.

lization cost of all links in the network:

$$\text{Minimize} \quad \sum_{\ell \in \mathcal{L}'} \kappa(\ell) \tag{4.1}$$

subject to all constraints explained below.

The first constraint assigns the utilization cost to all links based on the utilization of the link and the linear cost functions:

$$\forall \ell \in \mathcal{L}', \forall y \in \mathcal{Y}: \ \kappa(\ell) \geq y_a \sum_{f \in \mathcal{F}} \left(\frac{dem(f)}{cap(\ell)} \sum_{p \in \mathcal{P}} \rho(p) \cdot \mathcal{R}_\ell^p \cdot cor(f,p) \right) + y_b \tag{4.2}$$

On the right-hand side of this inequation, the summation over $p \in \mathcal{P}$ is just a Boolean indicating whether there is a path used for the considered flow f that traverses the considered link ℓ. If this is the case, we add the demand of the flow divided by the capacity

of the link to the utilization term, which is the summation over all flows $f \in \mathcal{F}$. Finally, the assignment of utilization cost κ to the link depends on all linear cost functions $y \in \mathcal{Y}$, i.e., a particular y generates a lower bound on the cost by multiplying the utilization with a constant y_a and adding another constant y_b.

The following constraint assures that each flow is routed via exactly one path:

$$\forall f \in \mathcal{F} : \sum_{p \in \mathcal{P}} \rho(p) \cdot cor(f,p) = 1 \tag{4.3}$$

SDNp requires again the two additional constraints on routing and the use of metric vectors, that have already been introduced in the model of Section 3.5:

$$\begin{gathered} \forall k \in [1,K],\ \forall p \in \mathcal{P},\ \forall d \in \overline{\mathcal{N}}_k : \\ dst(p,d) \cdot \rho(p) \leq \sum_{\vec{m} \in M_k} \varphi(\vec{m},d) \cdot cons(p,\vec{m}) \end{gathered} \tag{4.4}$$

$$\forall k \in [1,K],\ \forall d \in \overline{\mathcal{N}}_k : \sum_{\vec{m} \in M_k} \varphi(\vec{m},d) = 1 \tag{4.5}$$

4.3.2 Performance Evaluation

Comparison of the different operational schemes

Figure 4.2 shows the performance of load balancing in the Janos-US-CA topology in the form of histograms of link utilization, defined as the frequency of the occurrence of a particular link utilization value. The according experiments were carried out as follows: As initial scenario we used the traffic and link capacity values determined in the OSPF case of the previous experiment. The blue area in the figure depicts how OSPF utilizes the deployed links, which covers a wide range. This was again used as worst case result without any load balancing. We then used the routing optimization model detailed in Subsection 4.3 to balance the link loads such that the occurrence of

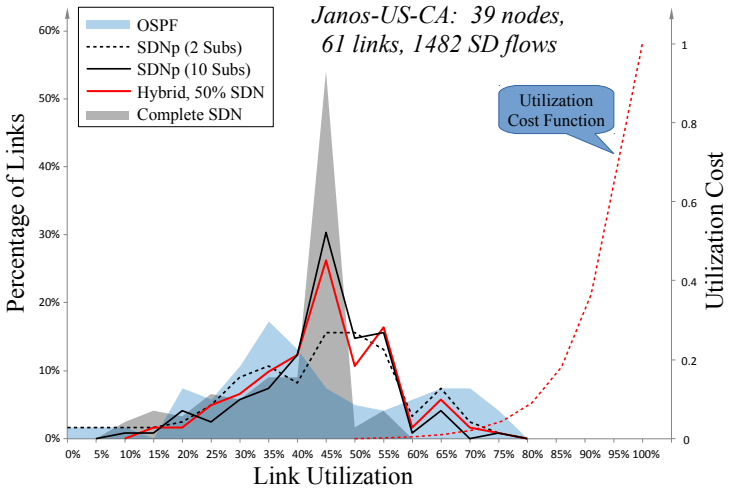


Figure 4.2: Link utilization histograms.

higher utilization degrees is less frequent. We again used the identical objective for the 50% (stacked hybrid) and the complete SDN deployment. The utilization cost function is superimposed in the figure (shown as the dotted red "Cost" plot). The optimality bound for load balancing is the case where all links are exactly equally utilized, which would result in a histogram with a single peak with 100% of links. Indeed, the histogram of full SDN deployment (plotted as gray area) exhibits a strong peak (54.1% of all links) right below 50% link utilization, which is exactly the upper bound of the "zero cost zone" for the objective function (i.e., links with $\leq$ 50% utilization cause zero cost in the routing optimization model). This result can be considered as the best case result for load balancing when routing is not constrained. Figure 4.2 also shows the histograms for SDNp with 2 (dotted black line) and 10 (solid black line) sub-domains. (Please note that due to clarity of this illustration we omitted the plots for

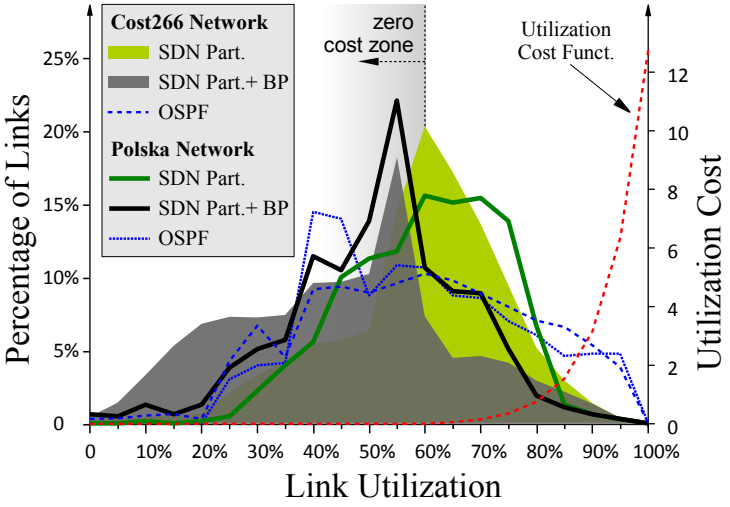


Figure 4.3: Percentage of links vs. utilization.

SDNp with 4 and 6 sub-domains, which however – if plotted in the same figure – would smoothly integrate between the plots for 2 and 10 sub-domains.) SDNp with 10 sub-domains allows for extensive routing control resulting in load balancing performance close to full SDN deployment, which indicates that a relatively small number of SDN-enabled routers (even compared to the stacked hybrid scheme with 50% SDN-enabled nodes, plotted as solid red line) can enable almost full traffic engineering capabilities in a network. The figure also shows that SDNp with only 2 sub-domains can already considerably improve the load distribution compared to regular OSPF operation.

Using optical bypasses in SDNp

Significant efforts in the community have been made to extend the paradigm of SDN to the area of optical transport networks and most "optics" vendors seem in a big rush to jump onto the SDN bandwagon. From the ongoing arguments on whether the extension of SDN to optical will provide an intelligent, automated, and unified (i.e., *the ultimate*) control plane, or whether an optical SDN is just the latest GMPLS reincarnation, it doesn't look like the die is already cast. However, a reasonable goal from the operations perspective is the capability to orchestrate the different layers with SDN. The fact is, by leaving all speculation aside, SDN and optical transport (in whatever flavor) can already support each other to a great extent. We here analyze the principles of SDNp extended with optical bypasses, i.e., optical circuit setup between pairs of SDN routers, as they are specifically chosen to create OSPF partitions. Optical bypasses extend SDNp to a higher degree of network control, and we show that these combined techniques can complement each other to a great extent. By allowing the provisioning of optical bypasses between the SDN-enabled sub-domain border nodes, traversed sub-domains can support bursty traffic and provide high resource utilization. On the other hand, OSPF routing protocol convergence issues remain unaffected, since optical bypasses are to be setup between SDN node pairs only, which (unlike regular OSPF routers) do not advertise the existence of such additional links through routing updates.

The performance of the proposed SDN partitioning with optical bypass is studied on two networks: i) Cost266 network, with 37 nodes and 57 links, and ii) Polska network with 12 nodes and 18 links, both taken from the SDNlib Library [36]. The network topology, SDN node placement (7 SDN nodes), and partitioning into sub-domains for Cost266 network is depicted in Figure 3.5b, and the same for Polska (3 SDN nodes) is shown in Figure 4.4. For each initial net-

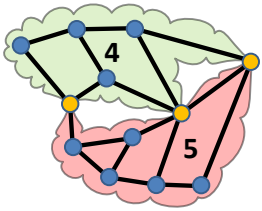


Figure 4.4: The Polska topology partitioned into 2 sub-domains.

work scenario, we assumed single path OSPF routing with uniform link weights (where in case of equal length paths the preferred one is randomly chosen). The chosen SDN node locations provide the partitioning with the highest number of inter-sub-domain flows. The general load balancing performance is assessed as a histogram of the percentage of links with a specific link utilization (averaged over multiple experiment runs). For each run, we first determine a (uniformly distributed) random traffic matrix (each random flow is in the range 0...7 Gbit/s) and compute the corresponding link loads, based on the assumed OSPF routing. We then assign the minimum link capacity (out of the set of 10 Gb/s, 40 Gb/s, 100 Gb/s, 400 Gb/s, and 1000 Gb/s) to each link according to its load. In every experimental run we assign new link capacities, whereby for every run the capacity chosen is the same for all analyzed schemes (OSPF / SDN Partitioning / SDN Partitioning + Bypass). In the OSPF scheme there is no minimum headroom reserved and no load balancing is performed.

After creating a basic OSPF scenario, in the second step (in which we start load balance the traffic) we assume that the network is partitioned with SDN nodes, like shown in Figures 4.4 and 3.5b. In the third step, we randomly[1] provision 5 optical bypasses among the

[1]The only restriction was that no bypass loops are allowed to avoid that bypasses heap up in some area.

7 SDN nodes in the Cost266 network (and 2 bypasses among the 3 SDN nodes in Polka, respectively). Finally, we superimpose the utilization cost function (including its zero cost landmark at 60%) in the histogram. First, we can observe in Figure 4.3 that SDN partitioning alone can improve the load balancing to a great extent: the number of highly utilized links (i.e., 80% to 100% utilization) is reduced significantly in both networks. Furthermore, provisioning only five optical bypasses ("SDN Part. + BP") in the Cost266 network and 2 bypasses in the Polska network allows to shift almost all traffic from highly utilized links below the zero cost landmark. By comparing the plots of the two networks with each other, it can be seen that the results look very similar, even though both networks are very different in terms of size, SDN nodes, number of sub-domains, and number of bypasses. It can be seen though that in the bypass-augmented topology of the Cost266 case, the amount of traffic shifted below the zero cost mark is larger that in the Polska network. This is caused by the increased number of routing alternatives in the larger Cost266 network, which allows a smoother load balancing than in smaller topologies.

One can claim that instead of using optical bypass or SDN, the network operator could be better served to either overprovision OSPF. To analyze this, let us focus on possible traffic churns in the Cost266 network and analyze the results shown in Figure 4.5. The process is the same as in the first experiment, but here we set the initial maximum OSPF link utilization to 90%, after which we increased 40% of all traffic flows (randomly chosen) by 70%. As it can be seen in the figure, the increased headroom of initial link capacities does not prevent congestion on a very high number (11.2%) of links. However, SDN partitioning (depicted as the green area) can already relief this scenario greatly. Most notable in this result is the direct comparison between "Full SDN" deployment (resulting in complete freedom of routing) and the proposed combination of SDN partitioning and op-

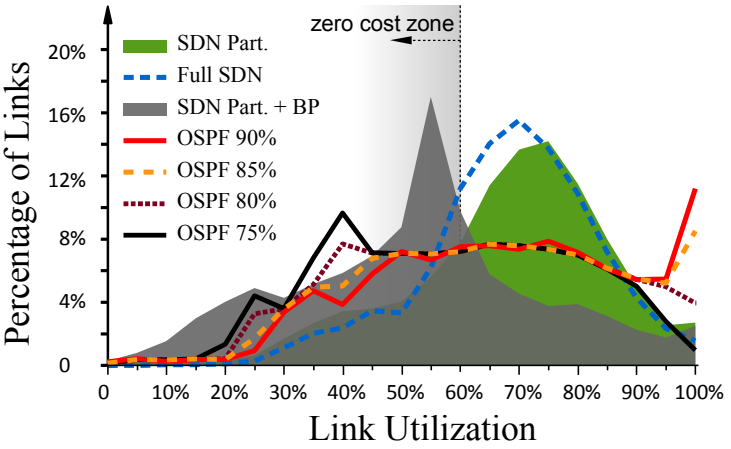


Figure 4.5: Adding more OSPF capacity reliefs congestion.

tical bypass, which leaves no doubt that a few optical bypasses are a much more valuable investment for network operators than equipping more nodes in the network with SDN capabilities.The figure further demonstrates that plain OSPF would require massive capacity increases to handle the discussed traffic increase: The figure plots the OSPF link utilizations for different maximums of the initial link utilization. A reduction to 75% can finally alleviate the congestion to a degree comparable to our proposed solution. However, reducing the maximum utilization from 90% to 75% required an increase of the total capacity of 24.4%.

4.4 Failure Recovery

Failure recovery is the ability of a control plane to dynamically react on failures in the network with an alternative valid routing (i.e., without routing loops and black holes) to avoid further traffic loss. Legacy routing protocols provide quick routing convergence,

but without taking the link utilization of the resulting routing into consideration, which typically requires a large capacity overprovisioning. Having central control on routing on the other hand allows to react on network failures with a rerouting that better utilizes the given link capacities. It is therefore interesting to analyze to what extent the capabilities of the compared control planes allow to avoid traffic overload in the network, which leads to a severe service degradation.

4.4.1 Mathematical Model

The mathematical model for failure recovery in SDNp, full SDN, and regular hybrid SDN/OSPF networks resembles almost completely the one of load balancing explained in the previous section. We will therefore limit the model details to the differences to the load balancing model. The objective of the failure recovery model is to react on link failures with the fewest possible routing updates, while at the same time avoiding over-utilization on any link. The two objectives are conflicting, which requires to trade routing stability off against balanced link utilization. This model resembles the previous one for the most part, because the mechanism basically load balances again, but here with a punishment on new link metric advertisements to keep the routing in the OSPF part of the hybrid control plane as stable as possible. However, routing changes through updates in the flow tables of the SDN border nodes are considered as invisible for OSPF and therefore do not provoke routing recomputation in legacy nodes. Therefore, our objective function resembles the one of the previous model, and additionally considers only the (stability) cost of metric changes (and not routing changes in general). We assume that, in case of a failure on link ℓ in sub-domain k, failure recovery is carried out as follows: All OSPF paths in sub-domain k that contain link ℓ are recomputed by the sub-domain's OSPF nodes (which is conform to regular OSPF operation), which in turn

changes the affected metric distance $\delta(r, b)$, and thus the mapping of the pre-computed metric vectors $\vec{m} \in M_k$ to the exit vectors $\vec{e} \in E_k$. All these parameters and the entire set of available routing paths $\mathcal{P}$ have to be recomputed as well for the here presented recovery model. The objective function is a minimization of the link utilization cost (like in the previous model) plus the (stability) cost for link metrics that are changed through the recovery process:

$$\text{Minimize:} \quad \sum_{\ell \in \mathcal{L}'} \kappa(\ell) + \sum_{\varphi} \varphi \cdot cost(\varphi) \tag{4.6}$$

We define the here used cost parameter $cost(\varphi)$ for the advertisement of metric vectors as the number of individual metric changes multiplied by a predefined punishment cost. In other words, all components of a metric vector $\vec{m}$ advertised for a specific destination through failure recovery, which are different from the metric vector advertised for the same destination before the link failure are counted and multiplied with some punishment cost. This punishment cost is then summarized over all metric advertisement variables φ and added to the objective function.

4.4.2 Performance Evaluation

Our failure recovery result is depicted in Figure 4.6 and shows (again in the form of link utilization histograms) to what extent the compared operational schemes can handle the occurrence of a sudden fiber cut in the network. This experiment was carried out based on the previous load-balancing scenario, where we simply deleted one link and reoptimized the routing. The results are averaged over all possible link failures in the network. For this experiment, we shifted the cost function to 80% in order to avoid only severely over-utilized links (which can easily occur in case of a fiber cut), while trying to minimize the number of rerouting events[2] in the OSPF part of

[2] As a rerouting event in the OSPF part of the control plan we consider each routing recomputation in an OSPF router due to a new LSA.

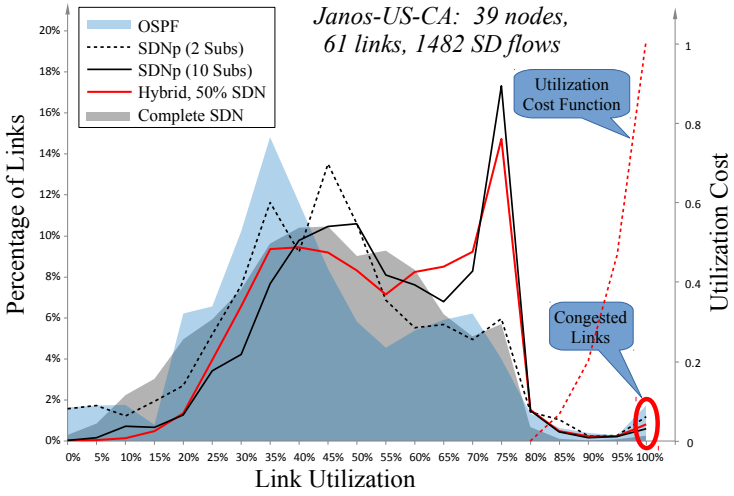


Figure 4.6: Link utilization histograms after a link failure.

the control plane. In other words, routing optimization using the here introduced mathematical formulation aims on fewest LSAs and reroutes flows only in case of over-utilized (or failed) links. The distribution of link utilization in case no routing reoptimization is possible – which is the case in OSPF – is shown as the blue area. Here, OSPF only assures that all nodes compute valid new shortest path routes, which is based only on link metrics and completely ignores the traffic load. Consequently, OSPF leads to the largest number of congested links. (See Table 4.2 for a numerical congestion comparison.) SDNp with 2 sub-domains (i.e., the control plane scheme with the weakest control on routing) is plotted as dotted black line and can already provide significant improvements compared to OSPF in terms of congestion. More sophisticated routing control is provided by SDNp with 10 sub-domains (solid black line) and stacked hybrid control with 50% SDN-enabled routers (solid red line), which both

lead to a significant peak at the lower cost bound at 80% link utilization on the one hand, and decreased congestion on the other hand. Full SDN deployment (plotted as gray area) can almost completely avoid congestion in our experimental set up. Please note that it is common in operative IP networks either to provide protection (i.e., idle backup links) or to overprovision link capacities such that congestion is avoided in case of link failures, which is expensive in terms of required capacity.

In the same experiment we measured the amount of excess traffic (i.e., packets that are dropped due to overloaded links) *after the control plane has finished all its routing reconfigurations* after a link failure. Note that traffic loss *during* routing reconfiguration is ignored in this table, as we assume that its duration is, firstly, very short compared to the duration of the link failure, and secondly, strongly depending on a wide range of topological and configurational parameters. The observed values are shown in Table 4.2, which additionally provides the average fraction of links that exhibit congestion and a measure on OSPF routing stability quantified by the number of actually performed routing recomputations in OSPF routers. Given our assumption of an initial link utilization threshold of 80%, a link failure in such an "economically" dimensioned OSPF network leads to drastic service degradation, which – on average – already entail 7.24‰ (i.e. more than 0.7%) of all packets dropped due to congestion (which also implies increased packet delays). Even though packet loss can not be avoided completely under these assumptions, the table shows that improved routing control can reduce the amount of excess traffic load significantly.

Routing stability in the OSPF part of a hybrid control plane is an important aspect, as each new LSA received by an OSPF router triggers a recomputation of the routing and forwarding table. The last column of Table 4.2 shows the average number of such OSPF reconfigurations for each examined control plane. Regular OSPF triggers

Operational Scheme	Traffic Loss	Cong. Links	OSPF Reconf.
OSPF	7.24‰	1.73%	78
SDN Partitioning (2 Subs)	3.57‰	1.19%	430.4
Stacked Hybrid, 50% SDN	2.47‰	0.81%	78
SDN Partitioning (4 Subs)	2.28‰	0.68%	110.5
SDN Partitioning (6 Subs)	2.03‰	0.60%	45.8
SDN Partitioning (10 Subs)	1.89‰	0.60%	24.7
Complete SDN Deployment	0.82‰	0.26%	0

Table 4.2: Service degradation after a fiber cut.

exactly 78 of such events after any link failure, as the used topology has 39 nodes and both adjacent routers advertise the topology change through flooding to the entire routing domain. The stacked hybrid control plane behaves identical, as all hybrid nodes perform regular OSPF below their SDN layer. However, as soon as all legacy nodes are substituted with SDN nodes, the legacy protocol can finally be turned off completely, which consequently results in zero OSPF reconfigurations for the case of complete SDN deployment. It can be seen from the last column of Table 4.2 that in case of SDNp the number of OSPF reconfigurations strongly depends on the number of sub-domains. Using this scheme with only two sub-domains provokes excessive use of OSPF reconfigurations, which however allows at least to half the amount of lost traffic compared to native OSPF operation. Due to the low number of SDN routers (4 out of 39) in this scenario, the capability to reroute traffic around congested areas (or failed links) by means of flow table updates from the central SDN controller is comparably limited. The only other

method to change routing in SDNp is to change the SDN border nodes used as sub-domain exit on a per-destination base, which in this case is heavily used by the failure recovery process to reduce packet loss and link congestion. It can however also be seen from the table that OSPF's routing stability increases rapidly in SDNp with an increasing number of sub-domains (and thus SDN nodes).

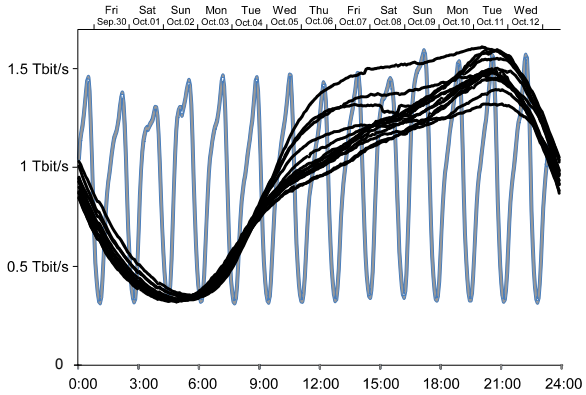


Figure 4.7: Daily traffic pattern (in Tbit/s) at DE-CIX.

4.5 Scheduling of Network Reconfigurations (Time Slicing)

In case frequent network reconfigurations are planned by the administrator to adapt the network to the actual traffic demand, e.g. load balancing, traffic engineering, or the so called *Load-Adaptive Energy Saving* (LAES), the scheduling of the individual tasks may has significant impact on the efficiency of the overall reconfiguration process. This section provides an analysis of the optimal scheduling of network reconfiguration cycles based on the example of LAES, whereas the made observations hold for other frequent network configuration tasks without loss of generality. The common idea of LAES approaches is to adapt the used link capacity to the traffic fluctuations, which would in turn allow to power down underutilized network resources under low traffic conditions. Considering the fact that Internet traffic follows a fairly stable daily pattern, there is a great opportunity to save energy by switching off idle networking equipment at the off-peak hours. (Please note that dynamically

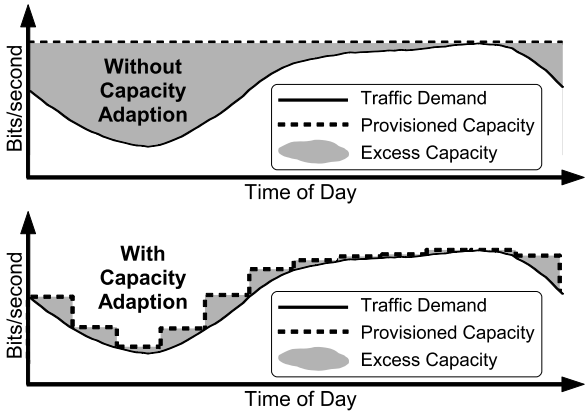


Figure 4.8: Relation between load and capacity.

switching off and on network equipment involves network operational challenges that are still unsolved.) Figure 4.7 [45] shows the typical traffic pattern that can be observed on backbone links of the Internet: The black lines demonstrate the stability of the daily pattern by superimposing 14 consecutive daily plots on a 24h time scale. Early approaches for LAES used simple day/night patterns, while others defined more granular (but uniform, e.g. 12×2 hours) time slices per day, thus allowing for more capacity adaptations, and thus better energy savings.

A typical time quantization scheme with uniform (i.e. equal size) time slices is shown in Figure 4.8, showing that after time quantization, the capacity graph becomes a step function. Its appearance depends on the number of slices and the start time of the first slice (which doesn't have to be midnight). Shifting a time slice is likely to entail a change of the maximum traffic value that occurs during that time slice, to which its capacity then has to be adapted to. Furthermore, it can be seen that some of the time slices in the graph could

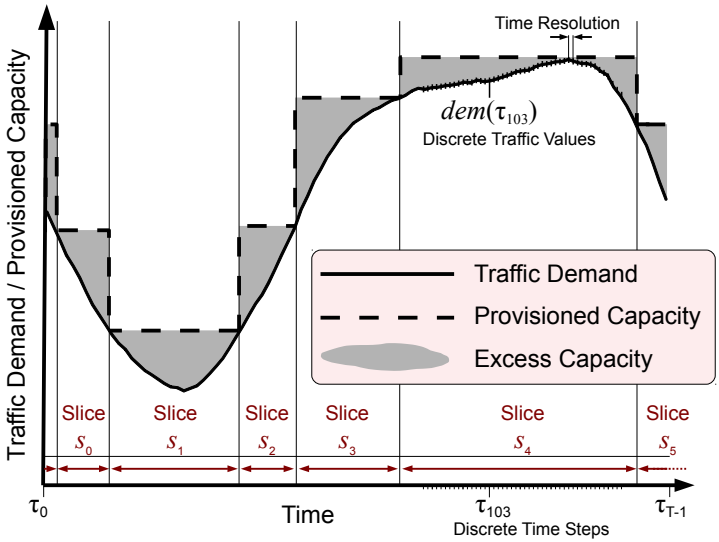


Figure 4.9: Capacity step function of a day partitioned into six time slices.

be merged without large effect on the provisioned capacity, resulting in fewer capacity adaption operations, which is what network operators usually prefer. In fact, there is a fundamental trade-off between the number of time slices and the achievable energy efficiency, versus the operational overhead in network management due to reconfigurations.

4.5.1 Analytical Model

Our time quantization approach is implemented as an Integer Linear Programming (ILP) model that minimizes the total excess capacity for a given number of time slices. The notation used is listed in Table 4.3. We assume that the input pattern of daily traffic demand

Parameter	Meaning
T	Total number of time steps per day
τ_i	The i^{th} time step
$\tau_0, \tau_{\text{T-1}}$	The first and the last time step
$dem(\tau)$	Traffic demand at time step τ
MaxD	Maximum traffic demand over the entire day
S	Total number of time slices per day
s_i	The i^{th} time slice
$s_0, s_{\text{S-1}}$	The first and the last time slice
MinDur	Minimum duration of every slice (in number of time steps)
Variable	**Meaning**
$\xi(s)$	Index i of time step τ_i, at which slice s begins
MaxD(s)	Maximum Traffic demand in slice s
$cap(\tau)$	Provisioned capacity at time step τ
GEQ(τ, s)	Boolean variable serving as *Greater Equal* operator, which is true only if $\tau \geq \xi(n)$
ELO(τ, s)	Boolean variable serving as *Element Of* operator, which is true only if τ is in slice s

Table 4.3: Notation for time slicing.

exists as an ordered array of discrete traffic values $dem(\tau)$, as illustrated in Figure 4.9 with τ_{100} and τ_{120}. Therefore, we consider time to be also discrete, with the same resolution as the traffic load (see "Time Resolution" at the top of the Figure), and a couple of discrete time steps are exemplarily depicted on the x-axis of the same figure. The time slices' duration must consequently be an integer multiple of a single time step defined by the traffic resolution. If for instance the resolution of the daily traffic pattern is 10 minutes (Figure 4.9), $dem(\tau_0)$ represents the traffic (in bits per second) between midnight and 0:10, $dem(\tau_1)$ represents the traffic between 0:10 and 0:20, and so on. In this regard, we denote τ as a time step and T the total number of time steps per day. In our example with 10 minute resolution, T would be $24\text{h} \cdot 60\text{min/h} \cdot (10\text{min})^{-1} = 144$. The 24 hours of the day are then partitioned into N slices (i.e. non-overlapping intervals). The provisioned capacity is assumed constant during any slice, and the capacity can be changed only at the transition from one slice to the next, so that the resulting capacity graph is a step function. Time quantization (i.e., the length and the starting time of all time slices) is sufficiently defined by the N time steps at which the N time slices begin. In order not to force the first slice to start at midnight, which would unnecessarily restrict time quantization, we allow the last slice to "overlap" midnight, so that it ends the day after it actually started. The goal of our ILP model is to set the slices' starting times such that the excess capacity is minimized. The objective function is to minimize the total provisioned capacity, i.e., :

$$\text{Minimize: } \sum_{i=0}^{T-1} cap(\tau_i) \tag{4.7}$$

subject to the following constraints:

For each time step τ, the capacity $cap(\tau)$ must be adjusted to the

maximum traffic value $\mathsf{MaxD}(n)$ of its corresponding time slice:

$$\begin{gathered}\forall\tau \in \{\tau_0, \ldots, \tau_{\text{T-1}}\},\ \forall s \in \{s_0, \ldots, s_{\text{S-1}}\} : \\ cap(\tau) \geq \mathsf{MaxD}(s) + (\text{ELO}(\tau, s) - 1) \cdot \mathsf{MaxD}\end{gathered} \tag{4.8}$$

The second term of the summation in unequation 4.8 ensures that when τ is not in slice n, the constraint is rendered inactive.

The maximum traffic value $\mathsf{MaxD}(s)$ must be constrained to be the maximum of all time steps during a slice s:

$$\begin{gathered}\forall\tau \in \{\tau_0, \ldots, \tau_{\text{T-1}}\},\ \forall s \in \{s_0, \ldots, s_{\text{S-1}}\} : \\ \mathsf{MaxD}(s) \geq \text{ELO}(\tau, s) \cdot dem(\tau)\end{gathered} \tag{4.9}$$

We now define the ELO variable, which mimics an *Element Of* operator: $\text{ELO}(\tau, s_i)$ must be forced to be 1, if τ is identical with or beyond the first time step of s_i, but not identical with or beyond the first time step of s_{i+1}:

$$\begin{gathered}\forall\tau \in \{\tau_0, \ldots, \tau_{\text{T-1}}\},\ \forall i \in [0, \text{N-2}] : \\ \text{ELO}(\tau, s_i) \geq \text{GEQ}(\tau, s_i) - \text{GEQ}(\tau, s_{i+1})\end{gathered} \tag{4.10}$$

Unequation 4.10 would fail for the last slice S-1, that overleaps midnight like Slice 5 in Figure 4.9 (i.e. it contains the final time steps of the day as well as the first time steps of the next day). The following unequation adapts to this special case:

$$\forall\tau \in \{\tau_0, \ldots, \tau_{\text{T-1}}\} : \quad \text{ELO}(\tau, s_{\text{S-1}}) \geq 1 + \text{GEQ}(\tau, s_{\text{S-1}}) - \text{GEQ}(\tau, s_0) \tag{4.11}$$

The second part of the *Element Of* operator definition is rather simple, as we can force $\text{ELO}(\tau, s)$ to be 0 (in case τ is not in slice s) by constraining each τ to be in only one slice:

$$\forall\tau \in \{\tau_0, \ldots, \tau_{\text{T-1}}\} : \quad \sum_{i=0}^{\text{S-1}} \text{ELO}(\tau, s_i) = 1 \tag{4.12}$$

We now define the GEQ variable, which mimics the *Greater Equal* operator: $\mathrm{GEQ}(\tau, s)$ must be forced to be to 1, if τ is identical with or beyond the first time step of s:

$$\forall i \in [0, \text{T-1}],\ \forall s \in \{s_0, \ldots, s_{\text{S-1}}\}:\quad \text{T} \cdot \mathrm{GEQ}(\tau_i, s) \geq i - \xi(s) + 1 \tag{4.13}$$

The factor T only assures that assigning 1 to GEQ is sufficient as long as the difference on the right hand side of the inequation is positive. We now need to force GEQ to be 0, if τ is neither identical with nor beyond the first time step of s:

$$\forall i \in [0, \text{T-1}],\ \forall s \in \{s_0, \ldots, s_{\text{S-1}}\}:\quad \text{T} \cdot \mathrm{GEQ}(\tau_i, s) \leq \text{T} + i - \xi(s) \tag{4.14}$$

Additionally, we need to constrain the ordering of time slices, so that $\xi(s_0) < \xi(s_1) < \ldots < \xi(s_{\text{S-1}})$:

$$\forall i \in [0, \text{S-2}]:\quad \xi(s_i) + \mathsf{MinDur} < \xi(s_{i+1}) \tag{4.15}$$

In this unequation, we introduce MinDur in order to allow the setting of a minimum length for all time slices, which may be desirable from the network operator's perspective. Again, the last time slice generates a special case for setting its minimum length, which requires the following constraint:

$$\text{T} - \xi(s_{\text{S-1}}) + \mathsf{MinDur} < \xi(s_0) \tag{4.16}$$

4.5.2 Performance Evaluation

We simulated an offline optimization in the NSFnet's expanded T1 topology (14 nodes, 22 links) from [46] along with the corresponding traffic matrix (linearly scaled up $\times 700$ to significantly utilize today's link capacities), and an original traffic pattern from the German Internet Exchange in Frankfurt (DE-CIX) like shown in Figure 4.7. The IP Port power model is shown in Table 4.4. The proposed quantization approach is compared with the equal size time slices

IP Link Bandwidth	2.5 Gb/s	5 Gb/s	10 Gb/s
Power Requirement	6	11	21

Table 4.4: Assumed power requirements.

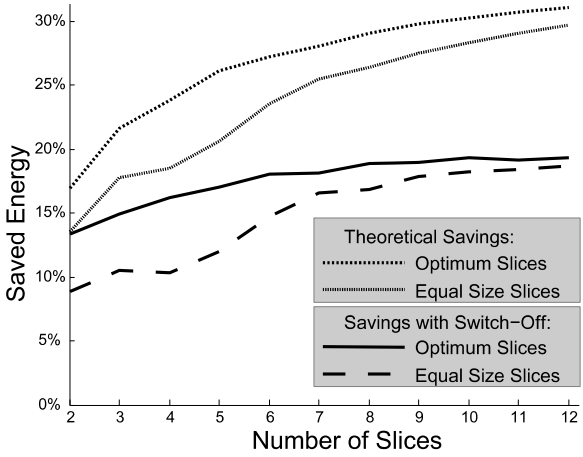


Figure 4.10: Time quantization and power consumption.

approach in the here presented energy saving benchmark. For the computation of the actual energy saving operations, we used the Switch-Off model presented in [30].

Figure 4.10 shows theoretical and actual capacity savings which can be obtained from time quantization proposed here vs. equal size time slicing. As expected, we can observe that optimum time quantization outperforms equal size time quantization in all cases, but the benefits of our method decrease with the increasing number of time slices. Due to the decreasing gradient of the graphs and the fact that both graphs already approach the upper bound of 35.8% (derived from 192 time slices, i.e. one capacity adaption at

every single time step), it's obvious from the figure that an increase of the number of time slices beyond 12 is unlikely to improve the results significantly. We also show the outcome of the network energy benchmark where considerable energy savings are possible in all analyzed cases, while the achievable energy savings are fairly below theoretical capacity savings. The cause of this phenomenon is twofold: First, in our network architectural model, energy does not scale linearly with capacity, since we assumed power requirements shown in Table 4.4, where we give a power "discount" for high capacity connections (i.e., it is "greener' to use a 10 Gbit/s port than aggregating two 5 Gbit/s ports). Second, with declining traffic load, the Switch-Off scheme successively reduces the connectivity of the network, leading to fewer options to optimize routing, and therefore resulting in less balanced link utilizations. We observed that Switch-Off would deliver results closer to the theoretical ones if the network topology initially had a higher nodal degree.

While the goal of the actual network energy saving benchmarking is to get a more realistic insight into the possible energy savings, it must be noted that the derived results still have to be considered as theoretical, due to some of the assumptions made for the network architectural model, such as zero switching time or the simple power model. Figure 4.10 shows that in case of equal size time quantization, partitioning the day into more than 9 slices doesn't increase the energy saving significantly. Compared to the proposed 12 equal size time slices in [47], our analysis shows that almost the same energy can be saved in case the number of time slices is reduced, which would reduce the operational overhead. We can also see that using only 6 optimized time slices results in almost the same energy saved.

Motivated by the previous result, we dedicate the next case study to the comparison of 6 optimum time slices and 9 equal size time slices. In Figure 4.11, we compare these two cases in terms of LAES operation efficiency, showing how much energy is saved depending on

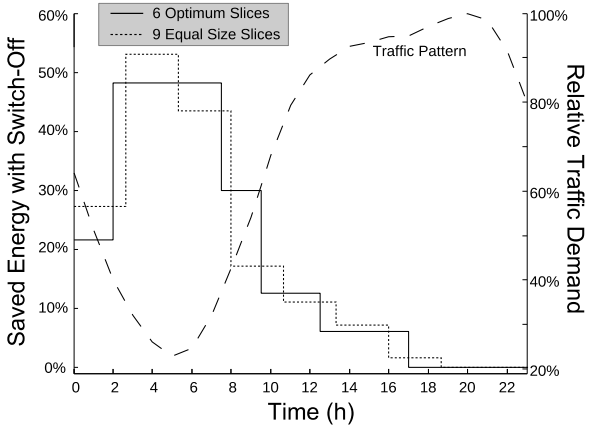


Figure 4.11: The saved energy for each time slice.

the time of day. The traffic demand pattern is depicted as the dashed line as reference, revealing that equal size slicing results in some energy saving operations at times when they are rather inefficient. For instance, slicing at minimum load, as is the case with the equal size slicing, is very inefficient, since both slices – the one before minimum load and the one after – have almost equal maximum traffic values (to which the capacity must be adapted), and therefore both slices could be merged to a single slice without losing notable savings, but resulting in lesser Switch-Off operations, and thus less traffic rerouting and less operational cost for reconfiguration of monitoring and fault management.

Figure 4.12 shows another comparison of the two methods in terms of potential network performance degradation. Packet delay performance depends on the transmission delay which in turn depends on the average IP hop count per transferred packet. The Switch-Off scheme results in an increased hop count (especially around 5am in our example). However, significant network performance degrada-

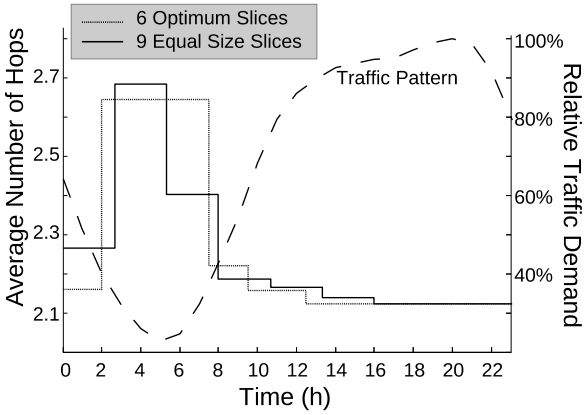


Figure 4.12: Average number of hops per packet with LAES.

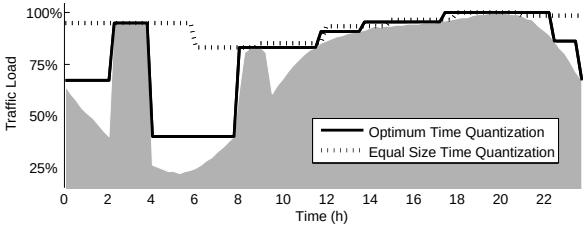


Figure 4.13: Adaptions to traffic surges.

tion in terms of delay (hop count) do not appear to be likely based on the results obtained.

All shown results strongly depend on the shape of the traffic pattern function. If, for instance, traffic were nearly constant throughout the day, the scheduling of the configuration actions become irrelevant for the efficiency of the actions. However, there are also traffic patterns possible, in which the here proposed time quantization approach would outperform equal size time slices significantly.

In order to study this phenomena, we altered the used original DE-CIX traffic pattern to the one showed as gray area in Figure 4.13 by adding two new traffic peaks at around 3am and 9am. Such a pattern could be caused e.g. by frequent large scale cloud data transfers at fixed times. However, we assume that such special cases would more likely affect smaller-size networks, e.g. due to server backups, then in IP backbone networks with typically larger port sizes. Since this is an hypothetical example, we only compare the theoretical capacity savings after time quantization with eight slices each. In this case, our proposed time quantization approach would yield 18.6% capacity savings (= 75.2% of the upper bound), while equal time slicing would only reach 6.7% capacity savings (= 27,2% of the upper bound). This extreme difference is caused by the fact that each new peak overlaps the border of two neighboring time slices in case of equal size time quantization. Hence, each peak must be considered in two time slices at times when the most capacity can be reduced, lowering the performance of this scheme substantially. Our approach on the other hand can easily adapt to such frequent traffic surges, so that it outperforms the equal size method almost three times over in this case. In summary, our approach shows significant benefits for bursty traffic patterns.

4.6 Traffic Monitoring

This section examines the operational aspects of the approach to solve the IP traffic matrix and the related monitoring problem in networks with a hybrid SDN/OSPF control plane, which is proposed in Subsection 3.4.2. We here address operational and implementation aspects of the required hybrid monitoring infrastructure. We discuss the timing issues of the measurements based on hands-on experiences in our lab, we explain the technical background and possible methodical pitfalls, we outline the design of our framework, discuss our

practical experiences with OpenFlow- and link-based measurements in our testbed, and address the conformity of our ideas with the novel paradigms in network management such as Network Function Virtualization (NFV).

There are multiple traffic matrix estimation techniques that use link loads and routing information, generally referred to as *network tomography*. Since the related linear system is ill-posed (and thus has multiple solutions), the accuracy of network tomography methods differ based on the statistical assumptions they make. One typical approach is to assume a certain traffic distribution function. Another method is to derive a solution with higher order statistics of the link loads, linear programming, or quadratic programming [48]. The *gravity model* [49] is another traffic matrix estimation technique initially developed for the research on road traffic. Here, the traffic matrix is derived only from the total traffic entering the network at each ingress and the total traffic exiting the network at each egress, whereas the interior network links and routing information are not considered. The gravity model can be used as input to the tomography method, which has been coined as *tomogravity model* [50]. Interested readers are referred to [51] for a detailed comparison of different traffic matrix estimation methods for legacy networks. We note that regardless of the method, all proposed traffic matrix estimations typically exhibit average errors in the range of 10% to 25% with some flow estimate errors above 100% [38].

Another method to obtain additional measurements is to periodically reconfigure the routing in the network. Paper [38] proposes rerouting by altering the IP routing protocol's link metrics in order to create an additional linear system $L = R \cdot F$ (containing different R and L). This new linear system can be combined with the original one to increase the rank. This method is performed repeatedly until the desired rank is achieved. The authors in [52] propose to route flows over fixed network monitor nodes. Please note that the here

presented approach does not require to alter the routing.

The adoption of SDN introduces additional traffic statistics that can be used to improve the estimation of the traffic matrix. [53] proposes to use the SDN-based measurements in addition to link counters to increase the rank of the estimation problem in data center networks. However, despite the assumption of a complete SDN deployment, the paper reasons that measuring every flow in the network is too costly. Consequently, a large-scale flow aggregation for the flow tables maybe required, which in turn results in a yet (not so) underconstrained linear system. [54] provides for the same purpose two efficient algorithms to determine measurement rules, but for hybrid SDN networks, assuming that TCAMs in the SDN switches do not suffice for all required monitoring actions. Contrary to the here presented approach, the current papers do not attempt to optimize SDN node placement to improve traffic monitoring.

The deployment of IP links between *all* ingress and egress router pairs (a so called full mesh topology) would allow the measurement of the complete traffic matrix only with SNMP-based link loads, which however does not scale, as the number of links would increase quadratically with the number of nodes. To address this issue, other standard layer 2 frameworks can be deployed for this purpose. For instance, MPLS could use LSPs, PBB-TE can provision E-Lines; even with the traditional Ethernet, VLANs could be configured and used for direct measurements of IE flows. Let us consider the case of MPLS as an example; here, the operator can set up an LSP between a pair of routers and install a packet counter on that LSP. In a network with N nodes, this would require the setup of $N \cdot (N-1)$ LSPs, and each LSP setup would involve the configuration of all routers along its routing path. Likewise with E-Lines and VLANs, the configuration overhead remains a concern.

The here proposed measurement scheme uses separate physical ports on a pair of IP routers, which have to be configured as a

backup to an already existing IP link. A backup link in addition to a regular IP link is easy to create and to configure, while it also allows measurements using regular SNMP link byte counters, which is vendor-independent and available in every router. In contrast to sampling, our method *directly measures* the IE flows using the SNMP link count on the backup link, so that an extrapolation from samples is not necessary. Additionally, we provide a solution for networks during the upgrade to SDN, when there are too few SDN nodes deployed, or they may be insufficiently located in the network.

The throughput of a link is determined based on consecutive queries of its byte counter, which is accessible with SNMP. This protocol is one of the most prevalent network management standards, developed by the IETF to allow the configuration and monitoring of network elements. It is defined following a manager/agent principle, where an SNMP-enabled network element (a router) implements an agent that can configure and monitor the network element and communicate with an SNMP manager. The protocol uses a hierarchical data structure called Management Information Base (MIB), which defines the syntax and semantics of the stored data. Its values are referred to as the MIB objects, and each object has a unique object identifier (OID). Routers commonly provide a MIB object for each port that provides counters for the incoming and outgoing bytes. The throughput of a link can then simply be calculated as the difference between two consecutive byte counter values divided by the time difference between the two queries.

Figure 4.14 shows a 4-node network to illustrate measurements on a backup link. The topology includes four routers, R1 to R4, and for the sake of simplicity, we only consider the six IE flows into the downward direction. The corresponding ill-posed linear system for the traffic matrix of this network has accordingly three rows (from the three original links) and six columns (from the six IE flows). The most beneficial backup link to use is obviously between R2 and

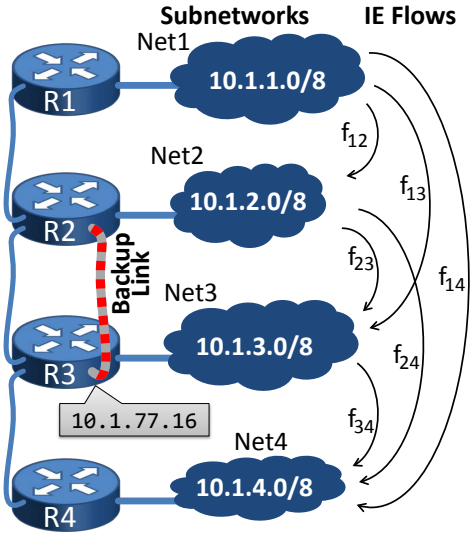


Figure 4.14: Possible measurements on a backup link.

R3, since the original IP link R2-R3 carries the most IE flows (f_{13}, f_{14}, f_{23}, and f_{24}). As in our example topology the rank of the linear system needs to be increased by three, it is sufficient here to measure only three of the possible four IE flows to let us solve the linear system of the traffic matrix. In other words, if we measure f_{13}, f_{14}, and f_{24} on the backup link and subtract the sum of their throughput from the load of the original link R2-R3, we obtain the throughput of f_{23}.

Figure 4.15 shows the basic configuration steps exemplarily for the command line interface (CLI) of Cisco IOS (in the blue boxes) and the timing of the corresponding retrievals of the SNMP byte counters from the backup link (red boxes): Assuming that the OSPF metric of the backup link is already configured to be larger than the one of

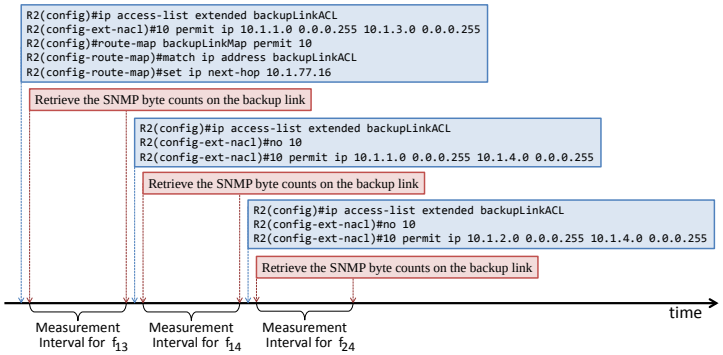


Figure 4.15: CLI configuration of router R2 in Figure 4.14.

the original link, we create an ACL with the name `backupLinkACL` on the backup link's ingress router R2. This ACL filters all packets between the subnetworks Net1 and Net3. Afterwards, we create a routing policy that forwards all those packets to the backup link. We now have rerouted f_{13} onto the backup link (that doesn't carry any other traffic), which allows us to measure it with two sequent retrievals of the SNMP byte counter on the backup link's port. In the second step, we reconfigure the ACL by deleting its previous matching rule and by adding a new one for flow f_{14} and measure its throughput on the backup link. Then, we similarly configure and measure f_{24} on the backup link. After all measurements are performed, the backup link could be decommissioned (if the layer 2 control plane supports port configuration) and the routing policy and the ACL can be deleted from R2.

The OpenFlow standard defines byte and packet counters for the entries of the flow table, which can be retrieved by the SDN controller. The effort required to retrieve the byte counters from OpenFlow devices in the network is comparably low, as all popular controller implementations provide some (REST-based) northbound in-

terface for management purposes. The throughput of a flow can then be determined – similar to a link's throughput – based on consecutive queries of its byte counter in the traversed OpenFlow router.

4.6.1 Statistical Behavior of Internet Traffic

Internet traffic (and thus each value of the traffic matrix that we aim to measure) is subject to two different sources of variability, which are 1) the changes due to the daily traffic pattern, and 2) traffic noise originated from the bursty nature of all the individual end-to-end flows[3] of which the highly aggregated IE flows consist. A complete cycle with sequential measurements has to be short enough (i.e. in the range of a few minutes), so that the impact of type-1 traffic changes resulting from the daily traffic pattern is small enough to be neglected. With other words, traffic changes due to the daily traffic pattern appear slowly and smoothly over the hours, while we assume that the time it takes to perform all necessary measurements is short enough to consider IE flows to be static.

Regarding the second source of traffic variability, it was shown in [55] that the noise fluctuations in short time scales can be viewed as a stationary random process with zero mean. Due to the effects of statistical multiplexing, the amount of noise of an IE flow is correlated with its size, such that small flows are dominated by noisy behavior and large flows can be characterized by a less noisy shape. This analysis was based on samples measured in Sprint's IP backbone and has shown that bursty traffic noise follows a power law with $\sigma = 1.56 \cdot \mu^{0.78}$. As we assume that aggregated traffic flows range from 2 Mbit/s to 20 Mbit/s, the standard deviation of an IE flow due to its noise can be expected to range from 3.9% to 6.4%.

[3] Please note that the term *end-to-end flow* here means a single communication stream according to Cisco's definition of the five-tuple <Protocol, Src. Address, Src. Port, Dst. Address, Dst. Port>.

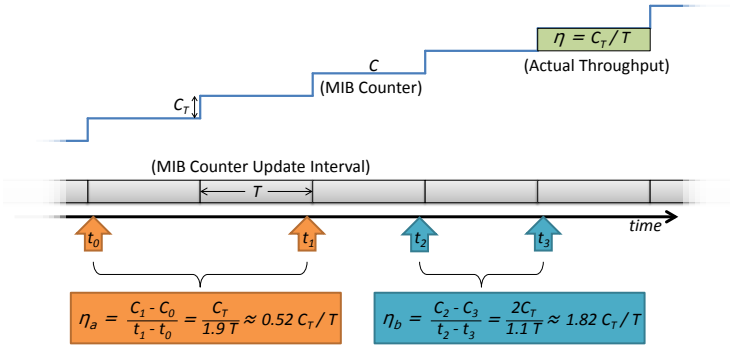


Figure 4.16: The MIB update problem.

In consequence, we suggest to consider type-2 measurement errors in the resulting traffic matrix and to limit the maximum number of sequentially measured flows per bypass, such that the total measurement period is not longer than five minutes, in order to keep the impact of the (slow) type-1 traffic changes negligible. The maximum number of sequential measurements does therefore depend on the minimum time of an individual measurement. Type-2 traffic statistics, on the other hand, can not be avoided and depend on the degree of aggregation in the network.

SNMP Timing Issues

Link byte counters are hardware implemented and not directly accessible from the outside. In order to provide that information via SNMP, the counter value is frequently written into the MIB, from where it can be requested. Apparently, the MIB is not synchronously updated with every tick of the counter (at least in the devices tested in our lab), but there is a fixed update interval P_{MIB}. Consequently, the calculation of the throughput of a link from its byte counter in the MIB is only straightforward when the measurement interval is

long enough: Two counter values C_1 and C_2 have to be retrieved, and the time instants t_1 and t_2 at the retrievals have to be known. The throughput η can then simply be calculated as follows:

$$\eta = \frac{C_2 - C_1}{t_2 - t_1}$$

However, this method requires that $t_2 - t_1 >> P_{MIB}$, i.e., the measurement interval has to be much greater than the MIB update interval. If this is not the case, the timing of SNMP requests must be synchronized with P_{MIB}. Figure 4.16 shows two extreme examples in which the calculated throughput differs from the actual throughput.

Figure 4.17 shows the MIB update rate (for a static throughput) of the different devices used in our testbed, which are: 1) an HP ProCurve E5406zl switch, 2) a Netgear GS108T switch, and 3) a Cisco 2811 router. The number of bytes transmitted over the network were recorded with Wireshark at the destination. The byte counters were requested simultaneously every 10ms from all network devices using SNMP. The MIB update frequencies are revealed by the step functions of the different counters. The HP switch updates every 500ms and the Netgear switch every 1000ms. The Cisco router updates the MIB by default only every 10 seconds (shown as Cisco R1 in the figure), but it can be configured (shown as Cisco R2) to do it at max every 10ms (using the not documented IOS command `snmp-server hc poll <interval>`).

The SNMP *response* time is the time between sending a request packet and the arrival of the corresponding response packet. It includes the network transfer time and the processing time of the device-internal SNMP agent. To understand whether SNMP queries with a relatively high frequency have impact on the packet processing performance of the equipment under load, we measured the response times of the devices in our lab, which is shown as a histogram in Figure 4.18. The system clock of the monitoring computer served

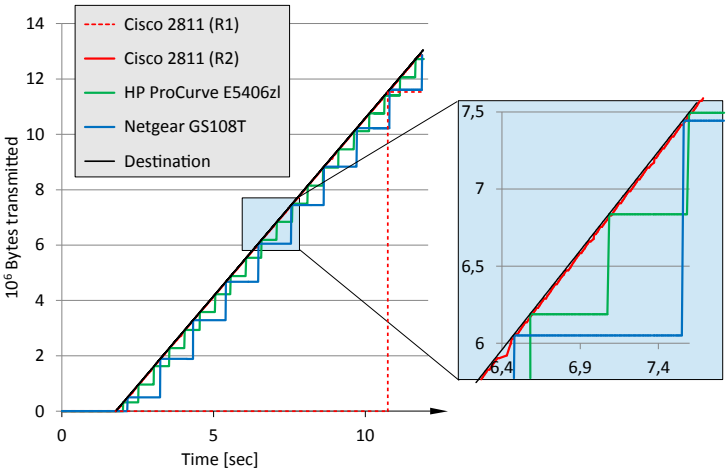


Figure 4.17: MIB update rates of the devices in our testbed.

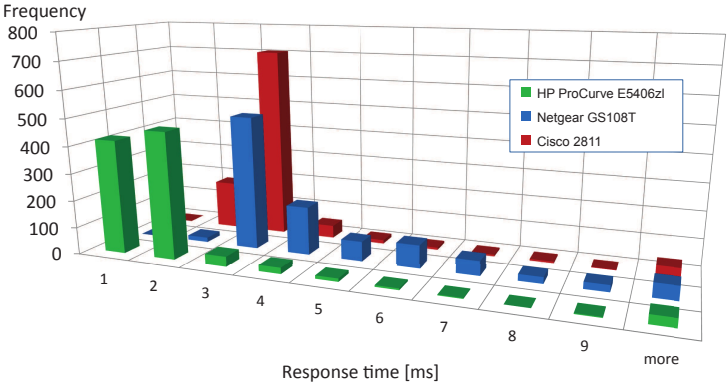


Figure 4.18: SNMP response times.

as reference time. 1000 requests were sent to each device and the figure shows that most response times are around 1-2 ms for the HP switch and mostly at 2-3 ms for the Cisco router. The Netgear switch is slightly slower and exhibits a larger variance in response times. Based on these observations, we consider all tested devices as suitable for fairly accurate throughput calculations (i.e. with measurement errors negligible for all practical purposes), assuming that the measurement time for each IE flow is an order of magnitude larger than the maximum response time of the devices.

Implementation Aspects

Operators of large IP backbone networks usually have a comprehensive suit of data bases and software tools for operation, administration and maintenance (OAM) for their infrastructure in place, which is commonly referred to as the Network Management System (NMS). Traffic monitoring is one of its important subsystems, which serves as source of information for many operational tasks, such as fault detection, capacity planning, anomaly analysis, etc. The monitoring subsystem required for the measurement scheme proposed in this subsection must implement 1) an interface to the SDN controller's northbound API to fetch the flow byte counters from the OpenFlow-enabled devices, 2) an SNMP manager to fetch the link byte counters from the legacy OSPF routers, and 3) an NMS-internal interface to the topology service subsystem (i.e. the data base that stores a model of the network topology including routing information and the specification of the resources).

As illustrated in Figure 4.19, we implemented a simple proof-of-concept monitoring application to perform traffic measurements from link and OpenFlow byte counters, which additionally allows to temporarily reroute specific flows to isolate them for measurements on a backup link. It retrieves OpenFlow counters from the central

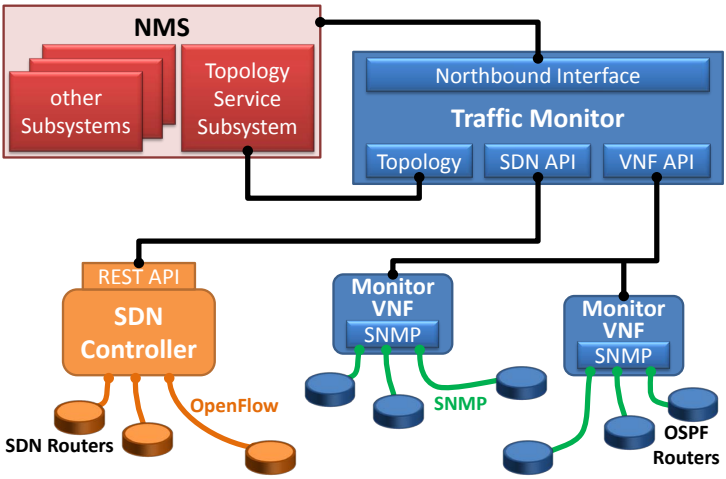


Figure 4.19: An architecture using monitoring VNFs.

SDN controller[4] through the controller's REST-based API. Please note that in large IP backbone topologies, the network transfer time for the SNMP counter requests, and more importantly its jitter, can become significantly large, which would affect the accuracy of the byte-counter-based measurements. We have therefore considered the deployment of the measurement application in the form of a parallelized Virtual Network Function (VNF) that can be operated in a distributed fashion, like shown in Figure 4.19. This would allow to implement the central part of the traffic monitor (the large blue box) as a module of the NMS, whereas the actual SNMP-based measurements would be performed by lightweight measurement VNFs (that could be operated on demand in virtual machines or application containers) closer to the actual devices from where the SNMP byte

[4]We used Floodlight [56] as SDN controller in our experiments, but other implementations (e.g. OpenDaylight) provide similar APIs, and could be used interchangeably.

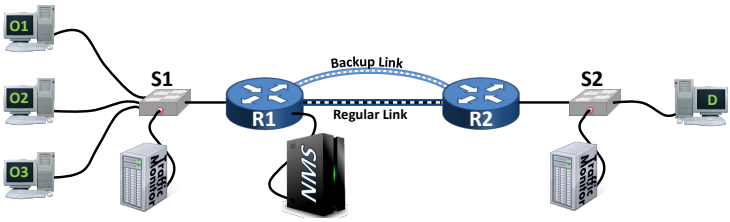


Figure 4.20: Testbed setup for our measurements on a backup link.

counters have to be retrieved. While our proof-of-concept application has not been entirely implemented in such a modular fashion, it is planned as future architecture for this research.

Testbed Measurements

We set up the testbed shown in Figure 4.20 to demonstrate SNMP- and OpenFlow-based measurements in our lab. The network consists of two routers, two Ethernet switches and a total of seven PCs. Four PCs represent endpoints of data connections: $O1$, $O2$, and $O3$, connected through a switch to router $R1$ serve as traffic origins, and D, connected to router $R2$, as destination. The routers are connected with two links: the first (i.e. working) link, denoted as backbone IP link, and the backup link. The fifth PC with the proof-of-concept NMS application is directly connected to $R1$. The incoming traffic at $R1$ and the outgoing traffic at $R2$ is measured via the intermediate switches' port mirroring function in order to obtain comparison values to the NMS measurements. We used the Iperf command line tool on the PCs to generate UDP traffic flows with constant bit rates. It should be noted that our testbed does not provide dynamic and automated port configuration, as the Layer 2 technology used is native Ethernet. In other words, the backup link setup is done manually by connecting the according devices with a patch cable. The proof-of-concept monitoring application accordingly lacks the

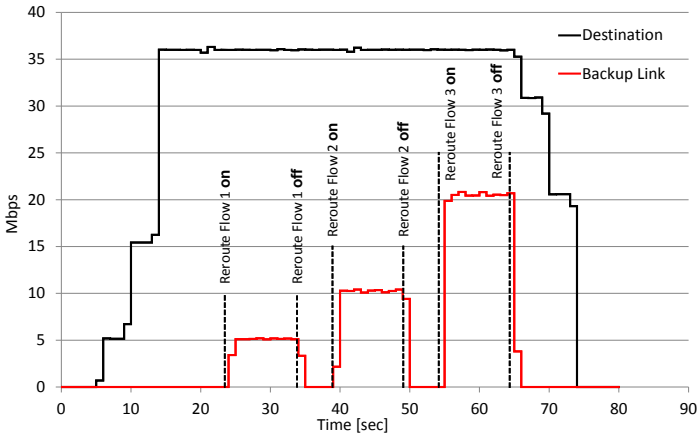


Figure 4.21: Bit rate measurements on the backup link.

ability to perform Layer 2 control actions.

Figure 4.21 shows the measured data rates (the three bursts of the red plot) in the first experimental setup using flow separation on the backup link. The aggregated traffic was captured (along with time stamps from the system clock) with the open-source packet analyzer Wireshark at the destination host (shown as the black plot in the figure). The three IE flows add up to approximately 36 Mbit/s at the destination (including the MAC frame and IP packet headers). The individual UDP flows were configured to have 5 Mbit/s, 10 Mbit/s, and 20 Mbit/s, which, along with the L2 and L3 protocol headers[5] add up to 35.9 Mbit/s, like measured at the destination. The byte counter of the backup link was requested every second and

[5]Ethernet frames have a size of 1518 bytes including the 18 bytes of the frame header (not including the Preamble and SFD) and the 20 bytes of IP packet header. Thus the overhead rate for the sent 35 Mbit/s IP payload is ca. 2.57%, resulting in ca. 35,9 Mbit/s measured with Wireshark at the destination.

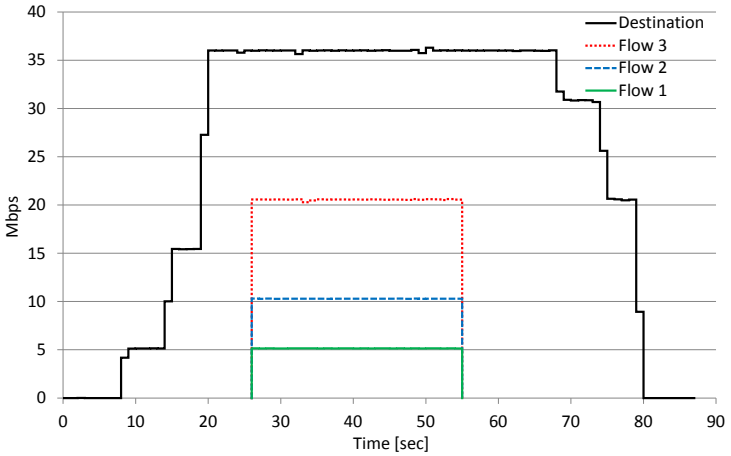


Figure 4.22: Flow bitrate measurements with OpenFlow.

the three flows were subsequently rerouted onto the backup link for ten seconds respectively. It can be seen that the measured throughput on the backup link matches the size of the three flows (plus the previously mentioned protocol overhead), which suggests that flow measurements on backup links are a practical solution to augment the monitoring system in place with additional traffic statistics.

Figure 4.22 shows the measurement for the same traffic scenario in a similar testbed setup, but with an OpenFlow switch in the center instead of the two OSPF routers. We retrieved the OpenFlow byte counters for the flow table entries of the three flows (again with a time resolution of one second) through the SDN controller, and the resulting throughput of the three flows is plotted as the red, blue, and green lines. It can be seen that, like in the case of SNMP-based measurements, the values calculated from the byte counters are again matching the configured UDP data rates, plus L2 and L3 overhead, similarly to the previous measurement.

4.7 Summary

We have analyzed, modeled, and discussed the OAM capabilities of hybrid SDN/OSPF networks in this chapter, and furthermore explained an efficient scheduling method for network reconfigurations that adapts to the network load. The provided mathematical models for regular (non-partitioned) hybrid SDN/OSPF and SDNp networks allow for the optimization of the important OAM tasks load balancing and failure recovery, which exhibit significant similarities as we have shown that failure recovery can actually be regarded as an extension of load balancing. A performance evaluation based on these models has shown that the operational performance of SDNp – again depending on the degree of partitioning – ranges between regular (non-partitioned) hybrid SDN/OSPF with 50% SDN nodes and full SDN deployment, which supports the numerical evaluation in the previous chapter.

The provided scheduling scheme in this chapter allows to more efficiently "slice" the operational day depending on the network's traffic pattern. We have shown that this can either significantly improve the performance of load-adaptive reconfigurations (like the exemplary demonstrated energy saving operations), or reduce the total number of actually required network reconfigurations without a loss on performance, which is especially desirable for network operators.

We have finally explained in this chapter the operational details for the novel monitoring approach that was proposed in Subsection 3.4.2. This approach allows to generate the IP traffic matrix from measurements of individual ingress-egress flows using both types of byte counters, from backup links between legacy routers and flow table entries of OpenFlow-enabled routers. Instead of using expensive monitoring infrastructure for non-SDN devices, we explained how to use PBR for backup ports and SNMP-based byte counters, features that are likely to be readily available in IP networks. We

showed that our method does not impact the IP routing in place, detailed the necessary configurational steps at the backup link ingress, and discussed SNMP timing issues. We also presented a software architecture for parallelized traffic measurements based on distributed VNFs that are connected to a central monitor, which allows to prevent timing-related measurement errors due to long transmission times in large network topologies. The experiences we made with our proof-of-concept implementation in our testbed confirm the applicability of our approach even in a networking environment containing outdated equipment.

5
Conclusion

A Trade-Off Between Centralized and Distributed

Distributed routing protocols like OSPF are still prevalent in IP networks due to their fault tolerance and ease of operation. SDN, on the contrary, is a novel centralized control paradigm, which recently became very fashionable in the networking industry and in academia. However, it's the decades of solid and reliable operation, what makes the distributed control protocols so desirable for network operators, who are thus reluctant to deploy SDN in operational networks. While centralization provides a comprehensive view and control over the global network state (which is a fundamental requirement for the most network optimization schemes), distributed protocols provide better scalability and robustness against failures. A predominant opinion seem to crystallize in this discussion, which recognizes the strong pros and cons in both, and that a hybrid control plane combining the advantages of both paradigms (mostly deployed in some overlay fashion) will be the best solution. However, a combination of both schemes in the same network will not automatically allow to cherry-pick from both worlds. Furthermore, in case an operator decides for a hybrid SDN/OSPF control plane, it was shown in this thesis that there is not the one solution that works equally well in all networking scenarios. Multiple parameters of the *hybridity*

have to be decided, which have significant impact on the required capital investment, the achievable degree of central control, the remaining amount of configuration autonomy of the legacy protocol, the complexity of network management, and the programmability and openness of the network for customized OAM applications.

In the simplest mode of hybrid operation, OSPF provides basic packet forwarding configuration for best-effort traffic, while the SDN controller can inject high priority rules for advanced and dynamic configurations, e.g. for load-balancing purposes, rerouting for QoS reasons, or for advanced failure recovery actions. This operational scheme requires OpenFlow-enabled OSPF routers that participate in the distributed routing protocol, but provide the central controller access to the configuration of their forwarding table. This operational mode can however easily lead to routing inconsistencies if both control planes are operated in the "ships-passing-in-the-night" mode, i.e. where both control planes are mutually unaware what the other one configures. It is therefore required that a higher management instance assures the consistency of both control planes. A more sophisticated hybrid SDN/OSPF approach deploys simple (i.e. non-OSPF) SDN routers that are configured to forward all OSPF protocol traffic towards the central controller, which in turn becomes responsible for all SDN-OSPF interactions. This implicates that the central controller establishes all OSPF neighborhood adjacencies and provides for all required protocol actions (i.e. frequent Hello packets, flooding of updates, response messages, etc).

SDNp as New Mode of Hybrid Operation

Both of the above mentioned hybrid schemes provide the same degree of control on the routing configuration in the network, which only depends on the number and location of SDN nodes. The performance evaluations based on the different planning and operational

models in this thesis have however repeatedly shown that the required number of SDN nodes in the network is relatively large. It was therefore proposed in this thesis to use SDNp as operational mode, i.e. to use the SDN nodes for an additional purpose: to partition the routing domain into sub-domains. It was shown how SDNp allows to steer to some extent OSPF's operations inside the sub-domains, which provides a new degree on network control.

It was explained in detail in this thesis, what the technical requirements are, and new mathematical models were provided that take into account the specific routing constraints of the different hybrid approaches for common network planning and management tasks. The here provided models cover the most important tasks, namely SDN node placement, SDN node deployment scheduling, capacity planning, provisions for a fault tolerant operation, load balancing, and failure recovery. Finally, we numerically evaluated the performance of SDN Partitioning in comparison to regular OSPF operation, full SDN deployment, and hybrid SDN/OSPF (assuming a 50% SDN deployment) without partitioning. Our results show that – depending on the degree of partitioning – SDNp provides network control capabilities between 50% and full SDN deployment, but with relatively few SDN-enabled routers. In the proposed scheme, the actual partitioning of the network (i.e., the adjusting of the size of the sub-domains) allows a trade-off of the degree of dynamic control (and thus the performance of the evaluated management operations) against carefreeness and routing stability. This claim is confirmed by our results: larger sub-domains provide less SDN control (due to more autonomous OSPF self-configuration), while smaller sub-domains increase the domination of the SDN control plane (and thus the performance of management operations that depend on dynamic routing control), but require a larger number of SDN-enabled nodes in the network. SDNp can finally be suggested as a convenient way of migration towards SDN, as the initial deployment of a few

SDN nodes already provides the partitioning of the routing domain into a few sub-domains, while additional SDN node deployments at later stages of the migration can be used to iteratively divide existing sub-domains into smaller ones, which will gradually increase the dominance of SDN.

SDNp Avoids Common Mistakes of Hybrid SDN/OSPF

As detailed in this thesis, the common mode of operation of a hybrid control plane either requires that a subset of the nodes are hybrid routers, i.e., routers capable of both OSPF and OpenFlow, or that the central SDN controller takes over all OSPF operations. Especially the former operational mode exhibits a number of design flaws. First, the FIB in a hybrid router is required to hold forwarding entries of OpenFlow *and* OSPF. As the FIB is commonly implemented in hardware to meet the requirements of interface line rates, it uses TCAM that can perform memory lookups in one clock cycle. This type of memory is known to be expensive, power hungry, and demanding in terms of silicon space [6]. As a result, hybrid routers are either provided with weakly dimensioned FIBs or are equipped with significantly more TCAM than OSPF *or* OpenFlow routers. In case of a hybrid controller (like with the latter regular hybrid mode or with SDNp), hybrid routers are not required, and plain OpenFlow switches are sufficient. The OSPF protocol is taken care of in the central controller and the SDN nodes forward all OSPF messages in a simple repeater mode: incoming messages from OSPF neighbors are forwarded through the OpenFlow channel to the central controller, and vice versa.

Second, in case of network failures the two control planes in the model using hybrid routers may remain unaware of each other's configurations for an unnecessarily long period of time, as mediation between both must be provided via the NMS. Failure recovery then

requires individual and sequential recovery processes, as the SDN part of the network has to wait for the OSPF part to completely converge before it can start its own routing optimization and reconfiguring. To make matters worse, both control planes may jointly cause forwarding anomalies, like routing loops and black holes [5], due to mutually inconsistent recovery actions. All of this is not the case in SDNp, as its hybrid control and management system is aware of the network status (topology, routing, and link utilization) in all OSPF sub-domains, as it receives the corresponding LSAs instantly in case of a failure. The complete visibility of the network allows SDNp to pre-calculate paths for network failures, although multiple failures remain a challenge also here.

Third, SDN-enabled routers need to be optimally located in a network with a regular (non-SDNp) hybrid control plane, which we have shown in Subsection 3.4.2 and showed (like many other studies referred to therein) that a poor deployment strategy can lead to a significantly lower performance. However, location optimization is not practical under dynamic traffic conditions, as new high priority traffic flows may not pass enough hybrid routers to provide sufficient traffic engineering capabilities. For fairness, SDNp can suffer from the same issue, especially in large sub-domains. However, SDNp can avoid this pitfall more easily: A flow with a long routing path is most likely traversing multiple sub-domains, and traffic engineering capabilities are provided in a per-sub-domain fashion. However, we note that SDNp – just like all other hybrid SDN/legacy control planes – suffer from an increased complexity of the network management, that needs to orchestrate two different control planes.

Bibliography

[1] Open networking foundation, members list. [Online]. Available: www.opennetworking.org/membership/members

[2] S. H. Yeganeh, A. Tootoonchian, and Y. Ganjali, "On scalability of software-defined networking," *IEEE Communications Magazine*, vol. 51, no. 2, pp. 136–141, February 2013.

[3] "Brocade advances SDN leadership with OpenFlow 1.3 support across IP routing and switching portfolio," Brocade press release, 2014, [Online]. Available: `http://newsroom.brocade.com/press-releases/brocade-advances-sdn-leadership-with-openflow-1-3--nasdaq-brcd-1094247#.Vl2FpkMQ1WI`.

[4] "PicOS overview," Whitepaper, 2014, pica8, Inc.

[5] S. Vissicchio, L. Cittadini, O. Bonaventure, G. G. Xie, and L. Vanbever, "On the co-existence of distributed and centralized routing control-planes," in *2015 IEEE Conference on Computer Communications (INFOCOM)*, April 2015, pp. 469–477.

[6] P. T. Congdon, P. Mohapatra, M. Farrens, and V. Akella, "Simultaneously reducing latency and power consumption in open-

flow switches," *IEEE/ACM Trans. Netw.*, vol. 22, no. 3, pp. 1007–1020, Jun. 2014.

[7] Gurobi Optimizer 5.0. [Online]. Available: http://www.gurobi.com

[8] J. Moy, "OSPF version 2," Internet Requests for Comments, RFC Editor, RFC 2328, April 1998. [Online]. Available: http://www.rfc-editor.org/rfc/rfc2328.txt

[9] D. Oran, "OSI IS-IS intra-domain routing protocol," Internet Requests for Comments, RFC Editor, RFC 1142, February 1990. [Online]. Available: http://www.rfc-editor.org/rfc/rfc1142.txt

[10] I. Pepelnjak, "Does centralized control plane make sense?" Ivan Pepelnjak's Blog at ipSpace.net, 2014. [Online]. Available: http://blog.ipspace.net/2014/05/does-centralized-control-plane-make.html

[11] C. Dixon, "On centralization in SDN and the applicability of OpenFlow," Colin Dixon's Blog at Cyberpunkture, 2014. [Online]. Available: https://blog.cyberpunkture.net/2014/06/on-centralized-sdn-and-openflow/

[12] M. Campanella, L. Prete, P. L. Ventre, S. Salsano, G. Siracusano, M. Gerola, M. Santuari, and E. Salvadori, "Bridging OpenFlow/SDN with IP/MPLS," poster presentation at the TERENA Networking Conference, 2014. [Online]. Available: https://tnc2014.terena.org/core/poster/10

[13] D. Levin, M. Canini, S. Schmid, and A. Feldmann, "Incremental SDN deployment in enterprise networks," *SIGCOMM Comput. Commun. Rev.*, vol. 43, no. 4, pp. 473–474, Aug. 2013.

[14] S. Agarwal, M. Kodialam, and T. V. Lakshman, "Traffic engineering in software defined networks," in *INFOCOM, 2013 Proceedings IEEE*, April 2013, pp. 2211–2219.

[15] F. Brockners, "Distributed? centralized? both?" Frank Brockners' Blog at Cisco Blogs, 2014. [Online]. Available: http://blogs.cisco.com/getyourbuildon/distributed-centralized-both

[16] B. Salisbury, "OpenFlow: Proactive vs reactive flows," Brent Salisbury's Blog, 2014. [Online]. Available: http://networkstatic.net/openflow-proactive-vs-reactive-flows/

[17] S. Jain, A. Kumar, S. Mandal, J. Ong, L. Poutievski, A. Singh, S. Venkata, J. Wanderer, J. Zhou, M. Zhu, J. Zolla, U. Hölzle, S. Stuart, and A. Vahdat, "B4: Experience with a globally-deployed software defined WAN," *SIGCOMM Comput. Commun. Rev.*, vol. 43, no. 4, pp. 3–14, Aug. 2013.

[18] B. Quoitin, C. Pelsser, L. Swinnen, O. Bonaventure, and S. Uhlig, "Interdomain traffic engineering with bgp," *IEEE Communications Magazine*, vol. 41, no. 5, pp. 122–128, May 2003.

[19] S. Vissicchio, L. Vanbever, and O. Bonaventure, "Opportunities and research challenges of hybrid software defined networks," *SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 2, pp. 70–75, Apr. 2014. [Online]. Available: http://doi.acm.org/10.1145/2602204.2602216

[20] S. Vissicchio, L. Vanbever, and J. Rexford, "Sweet little lies: Fake topologies for flexible routing," in *Proceedings of the 13th ACM Workshop on Hot Topics in Networks*, ser. HotNets-XIII. New York, NY, USA: ACM, 2014, pp. 3:1–3:7.

[21] S. Vissicchio, O. Tilmans, L. Vanbever, and J. Rexford, "Central control over distributed routing," in *Proceedings of the 2015*

ACM Conference on Special Interest Group on Data Communication, ser. SIGCOMM '15. New York, NY, USA: ACM, 2015, pp. 43–56.

[22] C. T. TechNotes, "Common routing problem with ospf forwarding address," Document ID 13682, 2005. [Online]. Available: http://www.cisco.com/c/en/us/support/docs/ip/open-shortest-path-first-ospf/13682-10.html

[23] J. K. Base, "Lsa exists in the ospf database, but not populated in the routing table," Article 13547, 2009. [Online]. Available: http://kb.juniper.net/InfoCenter/index?page=content&id=KB13547

[24] M. Chamania, M. Caria, and A. Jukan, "Achieving IP routing stability with optical bypass," in *Proceedings of the 3rd International Conference on Advanced Networks and Telecommunication Systems*, ser. ANTS'09. Piscataway, NJ, USA: IEEE Press, 2009, pp. 43–45.

[25] J. Moy, "Energy efficiency for network equipment: two steps beyond greenwashing," Whitepaper, Juniper Networks, Tech. Rep., 2010.

[26] S. Yang and F. A. Kuipers, "Traffic uncertainty models in network planning," *IEEE Communications Magazine*, vol. 52, no. 2, pp. 172–177, February 2014.

[27] C. J. Alpert and A. B. Kahng, "Recent directions in netlist partitioning: A survey," *Integration, the VLSI Journal*, vol. 19, no. 1-2, pp. 1–81, August 1995.

[28] E. Balas and d. C. C. Souza, "The vertex separator problem: a polyhedral investigation," *Mathematical Programming*, vol. 103, no. 3, pp. 583–608, 2005.

[29] B. Delaunay, "Sur la sphère vide. A la mémoire de Georges Voronoï," *Bulletin de l'Académie des Sciences de l'URSS*, no. 6, pp. 793–800, 1934. [Online]. Available: http://www.mathnet.ru/php/archive.phtml?wshow=paper&jrnid=im&paperid=4937&option_lang=eng

[30] M. Caria, A. Jukan, and M. Hoffmann, "A performance study of network migration to SDN-enabled traffic engineering," in *Global Communications Conference (GLOBECOM), 2013 IEEE*, December 2013, pp. 1391–1396.

[31] S. Verbrugge, "Strategic planning of optical telecommunication networks in a dynamic and uncertain environment," Ph.D. dissertation, Universiteit Gent, 2007.

[32] S. Türk, R. Radeke, and R. Lehnert, "Network migration using ant colony optimization," in *Telecommunications Internet and Media Techno Economics (CTTE), 2010 9th Conference on*, June 2010, pp. 1–6.

[33] C. Kronberger, T. Schondienst, and D. A. Schupke, "Impact and handling of demand uncertainty in multiperiod planned networks," in *Communications (ICC), 2011 IEEE International Conference on*, June 2011, pp. 1–6.

[34] S. Rajagopalan, "Adoption timing of new equipment with another innovation anticipated," *IEEE Transactions on Engineering Management*, vol. 46, no. 1, pp. 14–25, Feb 1999.

[35] B. Heller, R. Sherwood, and N. McKeown, "The controller placement problem," in *Proceedings of the First Workshop on Hot Topics in Software Defined Networks*, ser. HotSDN '12. New York, NY, USA: ACM, 2012, pp. 7–12.

[36] SNDlib library. [Online]. Available: http://sndlib.zib.de

[37] K. Papagiannaki, N. Taft, and A. Lakhina, "A distributed approach to measure IP traffic matrices," in *Proceedings of the 4th ACM SIGCOMM Conference on Internet Measurement*, ser. IMC '04. New York, NY, USA: ACM, 2004, pp. 161–174.

[38] A. Nucci, R. Cruz, N. Taft, and C. Diot, "Design of IGP link weight changes for estimation of traffic matrices," in *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, vol. 4, March 2004, pp. 2341–2351 vol.4.

[39] L. Hendriks, R. de O. Schmidt, R. Sadre, J. Bezerra, and A. Pras, "Assessing the quality of flow measurements from OpenFlow devices," in *8th International Workshop on Traffic Monitoring and Analysis (TMA)*, 2016.

[40] "Netflow performance analysis," Cisco White Paper, May 2007. [Online]. Available: http://www.cisco.com/c/dam/en/us/solutions/collateral/service-provider/secure-infrastructure/net_implementation_white_paper0900aecd80308a66.pdf

[41] G. R. Cantieni, G. Iannaccone, C. Barakat, C. Diot, and P. Thiran, "Reformulating the monitor placement problem: Optimal network-wide sampling," in *Proceedings of the 2006 ACM CoNEXT Conference*, ser. CoNEXT '06. New York, NY, USA: ACM, 2006, pp. 5:1–5:12.

[42] J. Altmann and L. Rhodes, "Dynamic netvalue analyzer - a pricing plan modeling tool for ISPs using actual network usage data," in *Advanced Issues of E-Commerce and Web-Based Information Systems, 2002. (WECWIS 2002). Proceedings. Fourth IEEE International Workshop on*, June 2002, pp. 143–148.

[43] M. Polverini, A. Baiocchi, A. Cianfrani, A. Iacovazzi, and M. Listanti, "The power of SDN to improve the estimation of the ISP traffic matrix through the flow spread concept," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 6, pp. 1904–1913, June 2016.

[44] B. Fortz and M. Thorup, "Internet traffic engineering by optimizing OSPF weights," in *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 2, 2000, pp. 519–528.

[45] "Traffic load statistics from the Frankfurt Internet Exchange DE-CIX, measured in october 2011 in a 15s time resolution," DE-CIX Competence Center Frankfurt.

[46] B. Mukherjee, *Optical WDM Networks (Optical Networks)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006, page 422.

[47] Y. Zhang, M. Tornatore, P. Chowdhury, and B. Mukherjee, "Energy optimization in IP-over-WDM networks," *Optical Switching and Networking*, vol. 8, no. 3, pp. 171–180, Jul. 2011.

[48] Y.Vardi, "Network tomography: Estimating source-destination traffic intensities from link data," *Journal of the American Statistical Association*, vol. 91, no. 433, pp. 365–377, 1996.

[49] D. Lam, D. C. Cox, and J. Widom, "Teletraffic modeling for personal communications services," *IEEE Communications Magazine*, vol. 35, no. 2, pp. 79–87, February 1997.

[50] Y. Zhang, M. Roughan, N. Duffield, and A. Greenberg, "Fast accurate computation of large-scale IP traffic matrices from link loads," in *Proceedings of the 2003 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer*

Systems, ser. SIGMETRICS '03. New York, NY, USA: ACM, 2003, pp. 206–217.

[51] A. Soule, A. Lakhina, N. Taft, K. Papagiannaki, K. Salamatian, A. Nucci, M. Crovella, and C. Diot, "Traffic matrices: Balancing measurements, inference and modeling," in *Proceedings of the 2005 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, ser. SIGMETRICS '05. New York, NY, USA: ACM, 2005, pp. 362–373.

[52] S. Raza, G. Huang, C. N. Chuah, S. Seetharaman, and J. P. Singh, "Measurouting: A framework for routing assisted traffic monitoring," *IEEE/ACM Transactions on Networking*, vol. 20, no. 1, pp. 45–56, February 2012.

[53] Z. Hu and J. Luo, "Cracking network monitoring in DCNs with SDN," in *2015 IEEE Conference on Computer Communications (INFOCOM)*, April 2015, pp. 199–207.

[54] Y. Gong, X. Wang, M. Malboubi, S. Wang, S. Xu, and C.-N. Chuah, "Towards accurate online traffic matrix estimation in software-defined networks," in *Proceedings of the 1st ACM SIGCOMM Symposium on Software Defined Networking Research*, ser. SOSR '15. New York, NY, USA: ACM, 2015, pp. 26:1–26:7.

[55] A. Soule, A. Nucci, R. L. Cruz, E. Leonardi, and N. Taft, "Estimating dynamic traffic matrices by using viable routing changes," *IEEE/ACM Trans. Netw.*, vol. 15, no. 3, pp. 485–498, Jun. 2007.

[56] "Floodlight," open-source SDN Controller. [Online]. Available: http://www.projectfloodlight.org/floodlight

Acronyms

ACL	Access Control List
API	Application Programming Interface
CLI	Command Line Interface
CMOS	Complementary Metal–Oxide-Semiconductor
DVFS	Dynamic Voltage and Frequency Scaling
ECMP	Equal Cost Multi-Path (Routing)
FIB	Forwarding Information Base (Routing)
GMPLS	Generalized Multi-Protocol Label Switching
IE	Ingress-Egress (Routing)
IETF	Internet Engineering Task Force
ILP	Integer Linear Programming (Optimization)
IP	Internet Protocol
IS-IS	Intermediate System to Intermediate System
ISP	Internet Service Provider
LAES	Load-Adaptive Energy Saving
LAN	Local Area Network
LSA	Link State Advertisement (OSPF)
LSDB	Link State Data Base (OSPF)
LSP	Label Switched Path (MPLS)
MAC	Media Access Control (Ethernet)
MIB	Management Information Base (SNMP)
MIP	Mixed Integer Programming (Optimization)
MILP	Mixed Integer Linear Program (Optimization)
MPLS	Multi-Protocol Label Switching

NE	Network Engineering
NFV	Network Function Virtualization
NMS	Network Management Systems
OAM	Operation, Administration and Maintenance
OID	Object Identifier (SNMP)
OSI	Open Systems Interconnection
OSPF	Open Shortest Path First
OTN	Optical Transport Networks
OXC	Optical Cross Connect
PBB-TE	Provider Backbone Bridge Traffic Engineering (Ethernet)
PBR	Policy-Based Routing
QoS	Quality of Service
REST	Representational State Transfer (API)
RFC	Request for Comments
ROADM	Reconfigurable Optical Add-Drop Multiplexer
SDN	Software-Defined Networking
SDNp	SDN Partitioning
SFD	Start Frame Delimiter (Ethernet)
SLA	Service Level Agreement
SNMP	Simple Network Management Protocol
SONET	Synchronous Optical Network
TCAM	Ternary Content-Addressable Memory
TCP	Transmission Control Protocol
TE	Traffic Engineering
UDP	User Datagram Protocol
VLAN	Virtual LAN
VNF	Virtual Network Function
WAN	Wide Area Network
WDM	Wavelength Division Multiplexing